

4

**AIR FORCE**



**AD-A218 204**

**HUMAN**

**RESOURCES**

**MAINTENANCE DIAGNOSTIC AIDING  
SYSTEM (MDAS) ENHANCEMENTS**

Garth Cooke  
Johnnie Jernigan  
Michael Huntington  
Theodore Myers  
Colleen Gumienny  
Nicola Malorana

Systems Exploration, Incorporated  
5200 Springfield Pike, Suite 312  
Dayton, Ohio 45431

DTIC  
ELECTE  
FEB 13 1990  
S E D  
Cc

**LOGISTICS AND HUMAN FACTORS DIVISION**  
Wright-Patterson Air Force Base, Ohio 45433-6503

January 1990  
Final Technical Report for Period May 1987 - March 1989

Approved for public release; distribution is unlimited.

**LABORATORY**

**AIR FORCE SYSTEMS COMMAND**  
**BROOKS AIR FORCE BASE, TEXAS 78235-5601**

## NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely Government-related procurement, the United States Government incurs no responsibility or any obligation whatsoever. The fact that the Government may have formulated or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication, or otherwise in any manner construed, as licensing the holder, or any other person or corporation; or as conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

The Public Affairs Office has reviewed this report, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This report has been reviewed and is approved for publication.

ROBERT C. JOHNSON, Branch Chief  
Combat Logistics Branch

BERTRAM W. CREAM, Technical Director  
Logistics and Human Factors Division

HAROLD G. JENSEN, Colonel, USAF  
Commander

# REPORT DOCUMENTATION PAGE

Form Approved  
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE January 1990	3. REPORT TYPE AND DATES COVERED Final - May 1987 to March 1989	
4. TITLE AND SUBTITLE Maintenance Diagnostic Aiding System (MDAS) Enhancements			5. FUNDING NUMBERS C - F33615-85-C-0010 PE - 63105F PR - 1710 TA - 00 WU - 16	
6. AUTHOR(S) Garth Cooke Theodore Myers Johnnie Jernigan Colleen Gumienny Michael Huntington Nicola Maiorana				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Systems Exploration, Incorporated 5200 Springfield Pike, Suite 312 Dayton, Ohio 45431			8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Logistics and Human Factors Division Air Force Human Resources Laboratory Wright-Patterson Air Force Base, Ohio 45433-6503			10. SPONSORING/MONITORING AGENCY REPORT NUMBER AFHRL-TR-89-13	
11. SUPPLEMENTARY NOTES				
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) <p>&gt; This report summarizes the research and development conducted to improve the effectiveness of the Maintenance Diagnostic Aiding System (MDAS). MDAS determines the next best action that should be taken to perform fault isolation and repair of a weapon system. This report provides information on key features that were added to MDAS such as the capability to suspend diagnostics; to provide a ranked, interleaved list of available test and repair actions; to diagnose multiple faults within a system; and to provide a summary of all actions taken during a troubleshooting sequence.</p> <p>- The research effort also showed that the modified split-half equation used by MDAS is as effective as the entropy equation in determining the information gained from a particular test. The ability for MDAS to generate logistic support data was also examined. The results showed that MDAS can record key information necessary to generate logistics data.</p>				
14. SUBJECT TERMS diagnostic advisory tool entropy equation diagnostic algorithm logistics analysis diagnostics split-half equation			15. NUMBER OF PAGES 62	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL	

## SUMMARY

The Air Force Human Resource Laboratory (AFHRL) is engaged in the research and development (R&D) of an Integrated Maintenance Information System (IMIS). This state-of-the-art demonstration system is capable of accessing and integrating several resources to provide data support for maintenance diagnostics. One functional asset of IMIS in the Maintenance Diagnostic Aiding System (MDAS) will utilize IMIS work station interfaces to access data bases such as the Reliability and Maintainability Information System (REMIS), Core Automated Maintenance System (CAMS), and Air Force Technical Order Management System (AFTOMS) for data to isolate and repair faults associated with failed aircraft components.

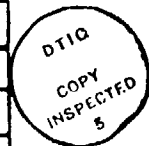
Initial efforts in MDAS produced a probabilistic diagnostic system for analysis of single faults in a generic aircraft environment. This report addresses software improvements to the baseline version of MDAS developed in earlier efforts, human interface issues, and new functions/capabilities for MDAS.

Work in software improvements have provided MDAS with added capabilities to evaluate multiple fault situations, Estimated Time In Commission (ETIC), and a rank interleaved list of tests and actions. In addition, software enhancements to increase the utility of MDAS have addressed the need for a human interface tool which provides technicians with the ability to suspend diagnostics, review previous actions, erase a test, show all tests/actions, and show all tests which could effectively attack a given symptom.

Initial human interface issues were addressed in the form of a "hands-on" field demonstration with a cross-section of potential users. Feedback gathered from this demonstration was helpful in identifying various human interface options. An improved prototype of the MDAS human interface was developed with use of Hypercard, a Macintosh software tool, for the actual coding of this interface. Additionally, several improvements in the capabilities of the MDAS software and collection of realistic validation data are provided.

The following functions/capabilities were examined for implementation in MDAS: wearout failure modes to accommodate the wearout phenomenon of newly installed aircraft components; numerous user dialogues; method of evaluating for a best test compared to the traditional information theory; and a feedback analysis tool that employs MDAS information to generate current logistical parameters for future diagnostic efforts.

<b>Accession For</b>	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	



## PREFACE

This report documents the findings and analyses conducted for the Air Force Human Resources Laboratory, Logistics and Human Factors Division, under the terms of contract F33615-85-C-0010, Task Orders 0010-03 and 0010-09.

Lt Dwayne Mason and Lt Randy Link, AFHRL/LRC, were extremely helpful in the research and development efforts to define and implement models and algorithms described in this report.

The research was performed by the Dayton regional office of Systems Exploration, Inc. (SEI). Principal investigators were Garth R. Cooke, Johnnie H. Jernigan, and Michael Huntington. They were assisted by Ronald J. Dierker, Colleen Gumienny, Nicola Maiorana, and Theodore Myers.

## TABLE OF CONTENTS

	<u>Page</u>
I. INTRODUCTION.....	1
Purpose.....	1
Background.....	1
Scope.....	2
II. RESEARCH AND DEVELOPMENT EFFORTS.....	2
Suspend Function.....	2
Multiple Faults.....	3
Human Interface.....	11
Log File.....	12
Estimated Time In Commission (ETIC).....	13
Review Previous Actions.....	14
Cannibalization Modeling.....	14
Aircraft Configuration.....	16
Interleaving Tests/Actions.....	18
Test Equipment Availability.....	18
Reinitialization/Change in Symptom.....	19
Feedback Analysis.....	20
Update Information Algorithm.....	27
Wearout Failure Modes.....	38
Model Repair and System Verification.....	42
User Dialogues.....	43
III. SOFTWARE TESTING.....	45
Subroutine Testing.....	45
System Integration Testing.....	45
IV. DISCUSSION AND CONCLUSIONS.....	45
REFERENCES.....	47
BIBLIOGRAPHY.....	48
LIST OF ABBREVIATIONS.....	49
GLOSSARY.....	50
APPENDIX: WEAROUT GRAPHS.....	52

# TABLE OF CONTENTS (Continued)

## LIST OF FIGURES

<u>Figure</u>		<u>Page</u>
1	Basic Operation of MDAS.....	1
2	Sample Multiple Fault System - Example 1.....	4
3	Sample Multiple Fault System - Example 2.....	8
4	Sample Fault/Test Relationship Matrix.....	10
5	Reduced Fault/Test Matrix.....	10
6	ETIC Table Display.....	14
7	MDAS Cannibalization Dialogue Flow.....	16
8	Aircraft Configuration Initialization Display.....	17
9	Sample Exhaustive Look-Ahead Output.....	22
10	Flow Schematic for the Sample Exhaustive Look-Ahead Output.....	23
11	Sample Log File Output.....	24
12	List of Log File Event Numbers .....	24
13	Feedback Analysis Example System A1.....	25
14	Feedback Analysis Example System A2.....	25
15	Feedback Analysis Example System B.....	25
16	Comparison Between Predicted And Simulated Actual Values.....	28
17	Feedback Flow from Base to Fleet Level.....	29
18	Binary Test Case Model.....	31
19	MDAS Formulation Isolation Tree.....	31
20	Entropy Formulation Fault Isolation Tree.....	32
21	Case I Model, 50 Plausible Faults.....	33
22	Case I Model, 4 and 10 Plausible Faults.....	34
23	Case II Model, 3 Implicated Faults Per Outcome.....	35
24	Case III Model, 2 Implicated Faults Per Outcome.....	36
25	Case IV Model, 1 Implicated Fault Per Outcome.....	36
26	MDAS vs Entropy for an Imperfect MTO.....	37
27	Binary Test Case.....	37
28	Failure Rate Distribution for a Population of Newly Installed Components.....	38

## I. INTRODUCTION

### Purpose

The purpose of this report is to document efforts to enhance the Maintenance Diagnostic Aiding System (MDAS) developed by the Air Force Human Resources Laboratory (AFHRL).

### Background

The basic objective of MDAS is to assist aircraft maintenance technicians in the isolation and repair of a fault or faults which have caused some symptom of improper system behavior. A key aspect of MDAS is that its algorithms are designed to minimize time to repair an aircraft, rather than time to isolate a fault. This philosophy takes advantage of the instances when rectification actions can be incorporated into the overall diagnostic sequence, resulting in the repair of faulty components in minimum time. MDAS employs special subroutines which modify split-half dependency models through the application of fault/symptom/component matching, component histories, probabilistic data, logistics constraints, and operational constraints. The basic operation of MDAS is described below and illustrated in Figure 1.

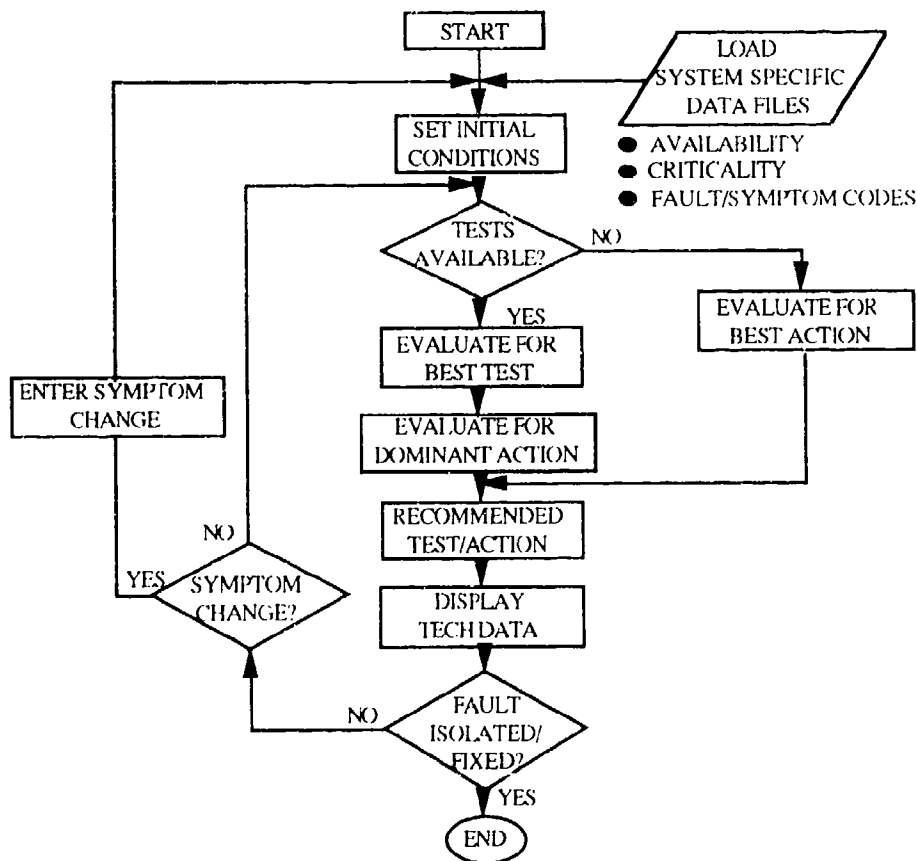


Figure 1. Basic Operation of MDAS.

After initialization, MDAS determines if any suitable diagnostic tests are available. Tests are then ranked based upon information gained per unit of invested time. Tests are then compared, utilizing time analyses (Dominant Action), to the most promising repair or replace activities to obtain the highest likelihood of fixing the problem in the least amount of time. A list of five ranked

actions is presented to the maintenance technician, the results of the technician-selected action are then used to reinitialize the plausible set of faults, and the sequence repeats. If no useful tests are exhibited and there is still ambiguity as to the fault location, the system then evaluates available actions based upon minimum time to rectification and presents its recommended action. This procedure continues until the fault is isolated or repaired.

### Scope

This report addresses identification, analyses, and development of characteristics associated with improved diagnostic capabilities and human interfaces of the MDAS.

The initial MDAS system (Cooke, Jernigan, & Treadway, 1986) was the baseline for this report. Three assumptions were used in the development of the initial version of MDAS:

1. One and only one fault exists in the system under test.
2. Parts obtained from base supply perform correctly.
3. Off-line sources of data (e.g., tests, workstation readings) are accurate.

With the subsequent development of efficient multiple fault handling capabilities in MDAS, the first and second assumptions were removed. This allowed MDAS to deal with "real world" maintenance diagnostic problems more effectively. Off-line sources of data are still assumed to be accurate throughout this task, but they may contain discrepancies in fault, symptom, test, and rectification relationships.

Other improved MDAS features which support diagnostic capabilities include: Estimated Time In Commission (ETIC); review previous actions; list all available tests/actions; cannibalization modeling; aircraft configuration; interleaving test/action; test equipment availability; reinitialization/change in symptom; feedback analysis; update information algorithm; wear out failure modes; log file; model repair and system verification; and show a symptom's test; and show all actions in sequence.

This report also addresses the implementation of several human interface features that allow MDAS to be used as an effective maintenance tool, adaptable to the maintenance technician's aptitude.

## II. RESEARCH AND DEVELOPMENT EFFORTS

### Suspend Function

This function was developed to deal with instances in which the diagnostic sequence needs to be temporarily halted (unavailability of parts or equipment, time constraints, etc.). The suspend function will allow the users to save their sequences and actions for use when they resume diagnostics. This saved information will include all aspects of the diagnostic evaluation, including the complete path taken to the point at which work was suspended. Typically, technicians would perform fault diagnosis using MDAS until they reached a point at which they could no longer continue. At this point, they would execute the suspend function, saving all their work. When ready to continue diagnostics, they would simply load the suspended file and return to the exact location from which they exited.

## Multiple Faults

### Multiple Fault Problem

The original version of MDAS was developed under the assumption that one and only one fault exists in the system being tested. There are obviously times when faults occur simultaneously. The purpose of incorporating a multiple fault capability into MDAS is to efficiently diagnose those instances.

### Definition of Multiple Faults

The term "multiple faults" refers to a situation when two or more faults occur simultaneously. Multiple faults can appear in a system in a variety of ways. The algorithms employed in MDAS are designed to handle all types of multiple faults that might appear, including:

Single Symptom. A single symptom is identified, but two or more failures might actually be present and causing that symptom.

Multiple Symptoms. Multiple symptoms are present which are caused either by a single fault or a combination of faults that completely "span" the symptoms.

### Approaches to Problem Solution

Two types of solutions to the multiple fault problem were considered: data adjustment and assumption relaxation. Data adjustment includes methods which represent multiple faults in the data set as fault states within the allowed set of faults. Combining multiple faults into fault states reduces the span of data to probable events, which facilitates analysis, data handling, and fault isolation. Assumption relaxation includes methods requiring more strict requirements be met before any fault is exculpated from the plausible set of faults. Faults that have been extracted from a plausible set of faults by a symptom change or repair by replacement will be stored in a data set which can be accessed as an unlikely event. The solution chosen for MDAS incorporates both approaches because no single method alone can solve the multiple fault problem.

### Multiple Fault Solution Development

The initial method used to handle multiple faults was based on the assumption that a probability could be assigned to each fault in the plausible set and single fault assumption methods could be applied for the times that a single symptom was encountered. Further, if multiple symptoms occurred and a single fault in the plausible set spanned each symptom (the intersection), then that fault would be the optimal place to begin analysis. If a single fault did not span the entire set of symptoms (no intersection), then multiple fault assumption methods would be invoked. These assumptions resulted in the mathematical method described below. This approach supports the initial assumption that the intersection is the best place to begin analysis. This solution is also supported by fault and test partitioning techniques which aid in the selection of a best test in a multiple fault situation.

## Multiple Fault Solution

Problem Definition. The following equations give examples of the multiple fault handling strategy of MDAS. Consider Figure 2, a multiple fault scenario including three symptoms and four faults:

	F1	F2	F3	F4
S1	.6	.2	.2	0
S2	.3	.7	0	0
S3	.1	0	0	.9

Figure 2. Sample Multiple Fault System - Example 1.

The probability that a specific fault caused a specific symptom is indicated by the numbers in the array. For example, the probability that F1 caused S1 is .6, and the probability that F4 caused S3 is .9. This can be extended to each symptom and fault combination. This model also shows how the faults span the set of symptoms.<sup>1</sup>

Mathematical Principles. The mathematical principle behind the method used to handle multiple faults is simple probability theory. Because the faults are considered independent events, if the probabilities of each event can be computed, then the probabilities of combinations of these independent events can also be computed. This follows from the fact that the probability of independent events' occurring simultaneously is the product of the individual probabilities of each independent event's occurring. Therefore, combinations of faults that could have caused the set of symptoms can be listed and the corresponding probabilities computed according to this rule.

Methodology. Once the probability of each combination is computed, the combinations are then rank ordered, with the one having the highest probability being the most likely to have caused the set of symptoms. Reduction of this initial plausible set of fault combinations is accomplished by removing redundant fault combinations from consideration. A redundant fault combination is a combination that is repeated in another combination. At this point in the analysis, the fault combination with the highest numerical product of individual fault probabilities is chosen from the remaining fault combinations. This is based on the assumption that the actions associated with rectifying each individual fault associated with a combination require equal amounts of time. Realistically, this is not the case. In order to make a more practical selection, this method incorporates the actual time required to rectify a plausible fault combination  $P(F_2F_4)$  as a single rectification  $P(F_3)$ . This is accomplished by factoring in time to rectification using a second step look-ahead approach. Therefore, removal of redundant combinations and incorporation of task times (time required to rectify faults in plausible combination) are the discriminating criteria. The rules incorporated in the multiple fault algorithm are as follows:

<sup>1</sup>F1 spans all three symptoms. This means that if F1 is known to have occurred, then all three symptoms will appear. Conversely, the appearance of only symptoms S1 and S2 would exculpate fault F1 from consideration because if F1 existed, all symptoms would be present. The symptoms are dependent on the faults, which are independent events. This idea provides the basis for the mathematical principles behind the method.

1. In removing redundant combinations, this method looks at the list of possible fault combinations and removes from consideration combinations that are included in a more probable combination. For example, if the method selected F2F4 as the most probable combination, any other combinations including F2F4 will be removed from the set because, by operating on and excluding F2F4, the 2 combinations will become invalid (given the same set of symptoms). This considerably reduces the plausible set. (See example section which follows)
2. A second method used to rank the reduced plausible set of faults is time to rectification. Once the plausible set is established, the probability of the combinations in the set is computed. This value is used in the following formulas in a second step look-ahead analysis. Given the plausible set of fault combinations  $PS=\{F1, F2\}$ , probabilities of each fault in the plausible set  $P(F1)PS$  and  $P(F2)PS$ , and the time to complete the action associated with F1 is A1 and F2 is A2, then the time to rectification for each possibility is given by:

$$T(F1)=A1+(1-P(F1))A1+P(F2)A2 \quad (1)$$

$$T(F2)=A2+(1-P(F2))A2+P(F1)A1 \quad (2)$$

$T(F1)$  is the time it takes to accomplish A1, plus the time it would take to undo A1 the percentage of the time A1 was the incorrect action, plus the time to complete A2 when A1 was incorrectly chosen first. The same logic holds for  $T(F2)$ . This analysis takes into account the probability that the chosen action will be correct and the length of time to complete associated actions. Additionally, from this method, one can easily compute the point that time will dominate over fault combination probability in the selection of the most probable combination. In summary:

1. Compute individual fault probabilities for n symptoms and m faults.  $P(F_j/S_i)$  is the probability that fault  $F_j$  has occurred, given that symptom  $S_i$  has been detected and the  $k$ 's are the individual symptoms.

$$P(F_j)_{j=1,m} = \frac{\sum_{k=1}^n P(F_j/S_k)}{n} \quad (3)$$

2. List possible fault combinations.
3. Compute probability of combinations of independent faults in each plausible set.

$$P(FC_i) = \prod P(F_j) \quad (4)$$

where  $P(F_j)$ s are the independent fault probabilities calculated in (3) which combine to make the fault combination  $FC_i$ . ( $i = 1$  to number of unique combinations.)

4. Rank combinations in order of highest probability and remove redundant choices (remove redundant  $P(FC_i)$ s).
5. Compute the probability of fault combinations as a probability of the reduced plausible set, allowing the sum of all  $P(FC_j)$  values in the reduced plausible set to equal one.

$$P(FC_i)_{PS} = \frac{P(FC_i)}{\sum_{i=1}^{nc} P(FC_i)} \quad (5)$$

where  $P(FC_i)_{PS}$  = probability of Fault Combination  $i$  of the reduced Plausible Set

$P(FC_i)$  = probability of Fault Combination  $i$  as computed in (4)

$nc$  = total number of combinations in reduced Plausible Set

## 6. Perform time analysis.

### Examples.

Example 1. For the multiple fault problem depicted in Figure 2, with all symptoms present, applying the above-method discussed results in the following computations and results.

#### 1. Fault probabilities:

$$P(F1) = \frac{P(F1/S1) + P(F1/S2) + P(F1/S3)}{3} = \frac{.6 + .3 + .1}{3} = 0.333$$

$$P(F2) = \frac{P(F2/S1) + P(F2/S2) + P(F2/S3)}{3} = \frac{.2 + .7 + 0}{3} = 0.300$$

$$P(F3) = \frac{P(F3/S1) + P(F3/S2) + P(F3/S3)}{3} = \frac{.2 + 0 + 0}{3} = 0.067$$

$$P(F4) = \frac{P(F4/S1) + P(F4/S2) + P(F4/S3)}{3} = \frac{0 + 0 + 0.9}{3} = 0.300$$

#### 2. Possible combinations: F1, F1F2, F1F3, F1F4, F2F4, F1F2F3, F1F2F4, F1F3F4, F2F3F4, F1F2F3F4

3. Combination probabilities:	<b>P(F1)</b>	<b>=</b>	<b>0.330</b>
	<b>P(F2F4)</b>	<b>= (.30)(.30)=</b>	<b>0.090</b>
	P(F1F2)	= (.333)(.30) =	0.100
	P(F1F4)	= (.333)(.30) =	0.100
	P(F1F2F4)	= (.333)(.30)(.30) =	0.030
	P(F1F3)	= (.333)(.067) =	0.023
	P(F1F3F4)	= (.333)(.067)(.30) =	0.007
	P(F1F2F3)	= (.333)(.30)(.067) =	0.007
	P(F2F3F4)	= (.30)(.067)(.3) =	0.006
	P(F1F2F3F4)	= (.333)(.30)(.067)(.30) =	0.002

4. Rank and reduce combinations from the list in Step (3). Bold entries indicate the combinations which are not redundant and comprise the reduced plausible set of fault combinations. In this case, F1 has a higher probability of being at fault (at this point in the analysis).
5. Probability of combinations in reduced plausible set:

$$P(F_A)=P(F1)_{PS} = \frac{P(F1)}{P(F1) + P(F2F4)} = \frac{0.330}{0.330 + 0.090} = 0.787$$

$$P(F_B)=P(F2F4)_{PS} = \frac{P(F2F4)}{P(F1) + P(F2F4)} = \frac{0.090}{0.330 + 0.090} = 0.213$$

6. Time analysis:

Case 1: all actions require equal time ( $A1=A2=A4=1$ )

$$T(F_A)=T(F1)=A1+(.213)A1+(.213)(A2+A4)=1.639$$

$$T(F_B)=T(F2F4)=(A2+A4)+(.787)(A2+A4)+(.787)A1=4.361$$

In this case, the option of F2F4 pays a penalty in time because it requires two separate actions of equal time.

Case 2: each combination requires equal time ( $A1=A2+A4=1$ )

$$T(F_A)=T(F1)=A1+(.213)A1+(.213)(A2+A4)=1.426$$

$$T(F_B)=T(F2F4)=(A2+A4)+(.787)(A2+A4)+(.787)A1=2.574$$

In this case, the time required for option F2F4 is lowered from 4.361 to 2.574 by cutting the time for that action in half. F1 is still selected under these conditions.

Under what time-related conditions will the intersection not be selected?

Case 3:  $A1=n(A2+A4)$  (let  $A2+A4=1$ )

$$T(F_A)=T(F1)=A1+(.213)A1+(.213)(A2+A4)$$

$$T(F_A)=T(F1)=n+(.213)n+(.213)$$

$$T(F_B)=T(F2F4)=(A2+A4)+(.787)(A2+A4)+(.787)A1$$

$$T(F_B)=T(F2F4)=1+(.787)+(.787)n$$

The point at which time will dominate can be found by equating the expressions for  $T(F1)$  and  $T(F2F4)$  and solving for  $n$ . In this case  $n=3.695$ . This means that in order for time to become a dominant factor, the action corresponding to the intersection must take at least 3.695 times longer than the action corresponding to the other combination, for this example.

Example 2. Consider a second example in which the probability that the fault lies in the intersection is small. This multiple fault system is depicted in Figure 3.

	F1	F2	F3	F4
S1	.1	.7	.2	0
S2	.1	.9	0	0
S3	.1	0	0	.9

**Figure 3.** Sample Multiple Fault System - Example 2.

1. Fault probabilities:

$$P(F1) = \frac{P(F1/S1) + P(F1/S2) + P(F1/S3)}{3} = \frac{.1 + .1 + .1}{3} = 0.100$$

$$P(F2) = \frac{P(F2/S1) + P(F2/S2) + P(F2/S3)}{3} = \frac{.7 + .9 + 0}{3} = 0.533$$

$$P(F3) = \frac{P(F3/S1) + P(F3/S2) + P(F3/S3)}{3} = \frac{.2 + 0 + 0}{3} = 0.067$$

$$P(F4) = \frac{P(F4/S1) + P(F4/S2) + P(F4/S3)}{3} = \frac{0 + 0 + 0.9}{3} = 0.300$$

2. Possible combinations: F1, F1F2, F1F3, F1F4, F2F4, F1F2F3, F1F2F4, F1F3F4, F2F3F4, F1F2F3F4

3. Combination probabilities:
- |                                      |              |
|--------------------------------------|--------------|
| <b>P(F2F4) = (.533)(.30) =</b>       | <b>0.160</b> |
| <b>P(F1) =</b>                       | <b>0.100</b> |
| P(F1F2) = (.10)(.533)                | 0.053        |
| P(F1F4) = (.10)(.30)                 | 0.030        |
| P(F2F3F4) = (.533)(.067)(.30)        | 0.011        |
| P(F1F3) = (.10)(.067)                | 0.007        |
| P(F1F2F3) = (.10)(.533)(.067)        | 0.004        |
| P(F1F3F4) = (.10)(.067)(.30)         | 0.002        |
| P(F1F2F4) = (.10)(.533)(.30)         | 0.002        |
| P(F1F2F3F4) = (.10)(.533)(.067)(.30) | 0.001        |

4. Rank and reduce combinations from the list in Step (3). Bold entries indicate the combinations which are not redundant and comprise the plausible set of faults. In this case, F2F4 has a higher probability of being at fault (at this point in the analysis).

5. Probability of combinations in reduced plausible set:

$$P(F_A)=P(F1)_{PS} = \frac{P(F1)}{P(F1) + P(F2F4)} = \frac{0.10}{0.10 + 0.160} = 0.385$$

$$P(F_B)=P(F2F4)_{PS} = \frac{P(F2F4)}{P(F1) + P(F2F4)} = \frac{0.160}{0.10 + 0.160} = 0.615$$

6. Time analysis:

Case 1: all actions require equal time ( $A1=A2=A4=1$ )

$$T(F_A)=T(F1)=A1+(.615)A1+(.615)(A2+A4)=2.845$$

$$T(F_B)=T(F2F4)=(A2+A4)+(.385)(A2+A4)+(.385)A1=3.155$$

In this case, F1 is selected even though its probability was lower. This results from the time penalty paid by F2F4 due to having two separate actions of equal time.

Case 2: all combinations require equal time ( $A1=A2+A4=1$ )

$$T(F_A)=T(F1)=A1+(.615)A1+(.615)(A2+A4)=2.230$$

$$T(F_B)=T(F2F4)=(A2+A4)+(.385)(A2+A4)+(.385)A1=1.770$$

In this case, the time required for option F2F4 is lowered from 3.155 to 1.770 by cutting the time for that action in half. F1 is not selected under these conditions.

Under what time conditions will the intersection be selected? Consider the following case.

Case 3:  $A1=n(A2+A4)$  (let  $A2+A4=1$ )

$$T(F_A)=T(F1)=A1+(.615)A1+(.615)(A2+A4)$$

$$T(F_A)=T(F1)=n+(.615)n+(.615)$$

$$T(F_B)=T(F2F4)=(A2+A4)+(.385)(A2+A4)+(.385)A1$$

$$T(F_B)=T(F2F4)=1+(.385)+(.385)n$$

The point at which the intersection will be selected can be found by equating the expressions for  $T(F1)$  and  $T(F2F4)$  and solving for  $n$ . In this case  $n=.626$ . In order for the intersection to be selected, the action corresponding to the intersection must take no more than .626 times the action corresponding to the other combination, for this example.

### Partitioning Faults and Tests Under Multiple Fault Conditions

One of the first actions MDAS completes is to evaluate available tests to determine the amount of information gained from each test, and which is the best. In a multiple fault problem, this process becomes complicated because a passed test and a failed test no longer have equal value. Consider the case in the first example. Figure 4 shows all combinations of tests that may be available to the technician.

	$F_A$	$F_B$
	F1	F2 F4
T1	0	0 1
T2	0	1 0
T3	0	1 1
T4	1	0 0
T5	1	0 1
T6	1	1 0
T7	1	1 1

Figure 4. Sample Fault/Test Relationship Matrix.

Tests associated with  $F_B$  will evaluate F2 or F4 independently, or as a combined set (F2F4). Evaluation of either fault independently or as a combined fault will result in the same information based on the analysis which showed that both F2 and F4 must be present for the three symptoms to have occurred. With this assumption, the span of tests can be reduced to  $T_A$ ,  $T_B$ ,  $T_C$  as shown in Figure 5.

$$T_A = \{T1, T2, T3\} = T1 = T2 = T3$$

$$T_B = \{T4\}$$

$$T_C = \{T5, T6, T7\} = T5 = T6 = T7$$

	$P(F_A) = .785$ $F_A$	$P(F_B) = .215$ $F_B$	Weight
$T_A$	0	1	$(.785)(.215) = 0.168$
$T_B$	1	0	$(.215)(.785) = 0.168$
$T_C$	1	1	$(0)(1) = 0$

Figure 5. Reduced Fault/Test Matrix.

This multiple fault system shows three tests spanning two faults. In this system, test groups  $T_A$  and  $T_B$  provide the best split of the faults. Since  $T_C$  completely spans both faults  $F_A$  and  $F_B$ , it is guaranteed to fail. Therefore, for this system, the only tests that guarantee information will be gained are test groups  $T_A$  and  $T_B$ . A pass on test group  $T_B$  will exculpate  $F_A$ , and a fail will implicate  $F_A$ . This is a significant departure from the methodology employed in the single fault assumption; consequently, a partitioning must be employed to effectively handle the occurrence of multiple faults.

Due to the unique situation presented in selecting a best test for a multiple fault problem, a special technique is needed to handle this increased complexity. The technique developed addresses the problem of tests guaranteed to fail (tests that span all fault combinations in a given symptom), ranks tests on the probability of fault isolation, and is used in conjunction with the multiple fault algorithms. MDAS has implemented the following technique, which constructs a partitioned set of probable faults and ranks tests to select a best test under multiple fault conditions. The first step is to partition the set of faults into fault combinations which have a high probability of having caused the symptoms by rank ordering the fault combinations in descending order. This

is easily accomplished by the multiple fault algorithms described earlier. This is the first partition. The second is created by grouping tests that provide the same information about a combination of faults ( $T_A = \{T_1, T_2, T_3\}$ ). These test groups can then be inserted directly into the existing best test formula.

The test group(s) with the highest score can then be selected as the best test(s). The best test(s) are then compared to available maintenance actions using the dominant action analysis. The results of this process minimize time to rectification and are used in conjunction with the normal diagnostic path until the problem is solved.

### Multiple Fault Conclusion

MDAS approaches the problem of multiple faults by considering several factors which include: distribution of fault probabilities for the symptoms being considered; how the faults span the set of symptoms; the lower probability of independent events' occurring simultaneously; and the influence of the time required to complete each possible action. Repeated analysis of multiple fault scenarios using this method showed that the intersection is not always the best place to begin. Probabilistic data and other information, such as time, will weight the fault combinations under consideration to determine the optimum starting point. This method supports one's intuition that the intersection is usually the best place to begin, and provides computational steps to confirm that belief. In order for the intersection not to be ranked as a best choice, the probability of the intersection fault must be smaller and/or the action time to repair that fault must be longer than other actions under consideration.

A technique has been developed to select a best test for a multiple fault problem. This technique eliminates tests which are guaranteed to fail, suspends from consideration tests which have unsure outcomes, and operates upon fault sets generated by the multiple fault algorithms.

### Human Interface

MDAS is intended to aid maintenance technicians in many ways, including for fault isolation and rectification, as a reference, and for training. As the system will be used by all multiple skills levels and is intended for application with a wide variety of systems, an effective human interface for the system is crucial. Several options for the human interface design were examined. The options examined during development were:

1. Option Presentation. At times during a diagnostic sequence, the "best" next activity is simply not practical; for instance, a requirement for an engine run during quiet hours, or a requirement to swap a part for which no replacement is available in supply. MDAS will recommend an alternative activity if the best activity is impractical. The interface question is whether the impractical best activity should be shown to the operator in its normal location as the "best" recommendation or should the alternative activity be shown?

2. Test Implications. As screen size permits, the entire subsystem undergoing diagnostics is shown to the operator as a connectivity block diagram, and the components containing plausible faults are highlighted. Also shown using different highlighting are the components which are examined by a particular test. Frequently, some portion of a test will examine components which are not plausible. The human interface question is whether to show on screen the non-plausible components examined by the test (accurate and complete) or just show which plausible components are examined (simple and uncluttered).

3. Text versus Graphic Presentations. The outputs from MDAS can be readily displayed as pure text, in a graphics format, or as a combination of graphics and text. A text display is much more representative of technical data displays contained in current Air Force Technical Orders. The human interface question revolved around acceptability and understandability of graphics versus combined text/graphics displays.

4. Flow of Control. MDAS can be used in three very different modes. It can be used simply as a "bookkeeper," which keeps track of actions taken and available actions remaining; it can be used as a test director, which presents only the computed next best activity with no other options possible; or it can be used interactively, which allows the maintenance technician to choose between the recommended action or some alternative. The human interface questions regarding these modes were: Which of these modes were more acceptable to the maintenance technician? Which, if any should be discarded from further consideration? Which should be the "default" mode for MDAS?

These human interface questions were examined during a field visit to Shaw AFB, South Carolina. Prototype displays for each of the above-listed human interface questions were shown to 18 maintenance technicians of various skill levels and specialties. Prior to the evaluations, the technicians were provided a briefing of MDAS capabilities and procedures. They were then given an opportunity to use MDAS to simulate evaluation of a fault in the F-16A Stores Management System (SMS). Each of the technicians was then asked to provide an assessment and indicate a personal preference from among the human interface options.

For the four human interface areas examined, the 18 users were overwhelmingly in favor of the following approach:

1. Show the "best" activity as it is normally ranked, and provide an indication that the activity is impractical through some sort of highlighting. The technicians can then use their own initiative and local knowledge to determine if they can overcome the factor which rendered the action impractical.

2. Show the full set of test implications. The system fidelity and the opportunity to learn about the system were felt to be more important than simplifying the graphics presentation.

3. Use a combination of extensive graphics and a limited amount of text for the MDAS displays.

4. Design diagnostics as an interactive task shared by MDAS and the technician. Both the "lockstep" and the "bookkeeping" modes were rejected by the evaluators.

### Log File

The utility to record major actions taken in a diagnostic sequence is called a log file. The log file will have numerous applications not only in MDAS, but also in the general maintenance area. The ability to record a diagnostic sequence will facilitate the recreation of that diagnostic sequence at a later date for either review or training purposes. The diagnostic sequence in a particular log file may be examined side-by-side with other sequences in order to compare diagnostic paths. Diagnostic sequences may be extracted from the log file to facilitate training activities, as examples or exercises for students. To further enhance support of the maintenance activities, diagnostic sequences in the log file can also be analyzed for information concerning the supplies, equipment, and manpower costs associated with specific diagnostic sequences and operating locations.

The log file will allow MDAS to be a more complete diagnostic tool and data capture device by providing information about actions taken during a repair session and allowing extraction of this information to update logistical parameters in the Integrated Maintenance Information System (IMIS) workstation interface environment. This ability to function in an integrated maintenance environment has shown utility in the development of a feedback analysis tool which will generate manpower, spares, support equipment, and other items of logistics concern.

The operation of the log file is fairly simple and straightforward. The log file is implemented using the major keystroke accumulator, which will log the actions taken and corresponding time and date. At the end of a diagnostic session, the information is written to an external file which can be accessed via the IMIS workstation.

#### Estimated Time In Commission (ETIC)

Estimated Time In Commission tables were developed so that the technician working on an aircraft would have the capability to determine the probability that the repair could be accomplished in a specific amount of time or, conversely, how much time would be required to achieve a specific probability of success. For instance, if a plane is recovered and fault codes are detected, the technician can input those codes and generate an estimated time to completion, in order to determine the probability that the plane can be fixed in time to complete the next sortie (if the next sortie is in 60 minutes), to fly later that morning or afternoon (within 120 minutes), or later in the day (within 240 minutes). This facility assists the technician and maintenance management personnel in making crucial decisions based upon the probability of when an airplane will be ready to fly.

The exhaustive look-ahead facility which was implemented in the baseline version of MDAS provides a comprehensive foresight to the evaluation of a plausible set of faults, and scans all paths to fault isolation. Using this facility of MDAS, ETIC tables can be created in two formats. The first provides probabilities of success against predetermined time windows (e.g., 30 minutes, 60 minutes, 120 minutes, and 240 minutes). The second provides time estimates against predetermined probabilities of success (e.g., 0.50, 0.75, 0.95, .99). These times were modeled for the following reasons: 30 minutes corresponds to a quick turnaround, 60 minutes corresponds to the next sortie, 120 minutes corresponds to the remaining half day, and 240 minutes corresponds to the next half day. The range of probabilities is arbitrary. These tables are generated according to the current state of the diagnostic process; therefore, these estimates will be unique to that state. Figure 6 illustrates how ETIC tables will appear in MDAS.

Time to Repair				
Time (min)	30	60	120	240
Prob. (%)	-	50	75	99

Probability of Repair				
Prob. (%)	50	75	95	99
Time (min)	60	120	200	240

Figure 6. ETIC Table Display.

#### Review Previous Actions

A complete diagnostic aid should give the user as much pertinent information and flexibility in operation as possible. In a typical diagnostic sequence, the number of actions required to identify symptoms and diagnose and repair faults may be many, and the time required to complete these actions may be great. In progressing through the diagnostic sequence, the technician may lose track of the actions he or she has completed or may wish to review actions already completed, as they may affect future diagnostic-related decisions. If the technician restarts a suspended session, he or she will want to review all actions previously accomplished in order to interact better with the diagnostic tool and its recommendations. Studies completed during research of human interface issues revealed that technicians desire access to as much information as possible about the diagnostic problem they are working. Therefore, the need for information regarding analysis of previous actions is sufficient to initiate research into methods to implement such a feature in MDAS. Such a feature has been researched and is called the review previous actions function. This function will allow the user to view all actions taken in the current diagnostic sequence.

#### Cannibalization Modeling

The first version of MDAS was able to differentiate between tests and actions that were feasible and those that were not because of non-availability of necessary parts. In this preliminary version, if the MDAS algorithms selected an option that could not be accomplished because of unavailable parts, the option was still displayed, but in reverse video, denoting parts that were not available. A technician selecting this option would receive a warning that the needed parts are not available from supply, but would be allowed to proceed. By modeling the steps the technician performs in cannibalization and the associated technical order requirements, several facets of the maintenance process are expedited: (a) Crucial maintenance can be performed to bring a system to operational status even in the wake of supply shortages (e.g., combat); (b) functional testing is facilitated to promote fault isolation; and (c) considerable time may be saved by shortcutting the traditional supply process.

## Cannibalization Process

Cannibalization is when the technician removes "good" parts from one aircraft for use in another which is under repair. This action can be likened to a swap action, the major difference being that the swapped part is coming from another plane rather than from supply. During this "swap" action, technical orders are needed for removal and replacement of parts for the plane being cannibalized as well as the plane under repair. MDAS will assist in this process by guiding the technician through the cannibalization, providing appropriate technical orders in a logical order, and allowing the technician to choose how the cannibalized part is to be used.

Two cases can arise during the cannibalization process. The first is one where the swapped part is to remain in the aircraft under repair. In this case, there is a pressing need to bring the plane under repair to operational status, and disabling another plane is an acceptable consequence. The second case occurs when the swapped part is needed only for troubleshooting purposes. In this case, the swapped part will ultimately be returned to the plane from which it was taken. Each case requires a different sequence of actions by the technician.

## Cannibalization Dialogue

In order to incorporate this facility into MDAS, a dialogue "tree" of possible paths the technician might take during cannibalization was developed and used as the baseline for implementation. This dialogue includes the steps associated with each possible cannibalization case and queries the user as to his or her intent in order to display the appropriate technical orders. The cannibalization routine is initiated when the user selects an option that is displayed as unavailable. When this occurs, MDAS displays a warning stating the selected option is unavailable, and asks if the user wishes to cannibalize another aircraft for the necessary part. After the warning screen, the user may choose to initiate the cannibalization sequence with a YES response and progress through the shown sequence, or the user may avoid cannibalization with a NO response, thus returning to the place where the unavailable option was selected. If the user chooses to cannibalize another aircraft, three sets of tech orders are displayed in sequence for removal of the "good" part from the cannibalized aircraft, removal of the "bad" part from the plane under repair, and replacement of the good part in the aircraft under repair. After installation of the good part on the broken aircraft, a functional test is performed to evaluate the impact of the good part. If the functional check fails, further removal and replacement of parts is suspended until the system checks out. Once the system check is OK, tech orders are displayed to return parts to their original location. If the functional test passes, MDAS provides a prompt to determine if the user intends to use the cannibalized item for troubleshooting or as a permanent fix. Depending on the response, MDAS displays appropriate tech orders to facilitate that choice. When any one of these three paths is completed, diagnostics are resumed. This cannibalization process is shown in Figure 7.

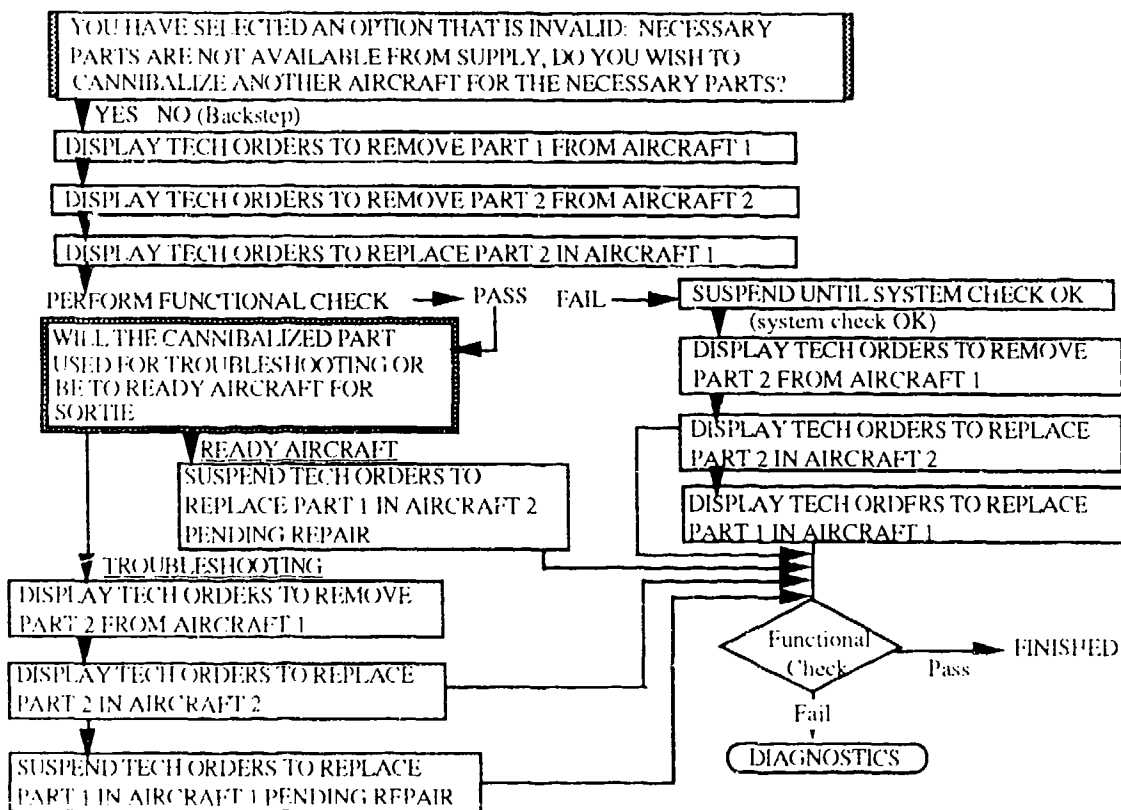


Figure 7. MDAS Cannibalization Dialogue Flow.

### Aircraft Configuration

System configuration is an important consideration for any diagnostic aid because as configurations change, the set of valid plausible faults will also change. Therefore, if the diagnostic aid is unaware of changes in system configuration, it will continue to diagnose a system in a generic manner, resulting in faulty or clouded results. An intelligent diagnostic aid is one which obtains and evaluates numerous inputs specific to a problem. System configuration is an input specific to every problem. If this information is not forwarded to the diagnostic aid, symptoms may appear which are "normal" for a specific configuration. For example, consider an F-16 which is configured completely conventional but has a nuclear Remote Interface Unit (RIU) installed on one of the pylons. One of the symptoms that will appear is lost communication with the nuclear RIU. However, this error message is normal for the all-conventional configuration of the F-16. Therefore, due to configuration irregularities, a symptom is present that is normal for the current configuration.

One way of avoiding such confusing circumstances is to flag these symptoms, excluding the implicated faults from the plausible set and thus making the diagnostic sequence more efficient. This aircraft configuration feature allows the technician to tell MDAS how the system being diagnosed is configured, which will expedite diagnostics by removing faults not possible with the specified configuration.

This feature is demonstrated on the F-16 SMS, but is valid for all systems. The individual stations on the F-16 can be configured, nuclear, conventional, or blank (nothing on the pylon), depending on which remote interface unit (conventional/missile or nuclear) is within a pylon. Each pylon can have more than one type of RIU installed, but only one can be active at any given time.

MDAS can use this configuration information to remove, prior to diagnostics, any faults from the plausible set associated with symptoms that do not correspond to the specified configuration. For example, consider Station 3 on the F-16. This station is capable of carrying either nuclear or conventional stores or it can be void of stores. If the aircraft is carrying conventional stores connected to a conventional RIU, the SMS will show a fault code identifying lost communication with the nuclear RIU. Should the technician pursue this fault code, he or she would be wasting time because that code is normal when Station 3 is carrying conventional weapons and both a Nuclear Remote Interface Unit (NRIU) and Conventional Remote Interface Unit (CRIU) are installed within the pylon. Consider a second case in which the same RIUs are not connected to any stores and the aircraft is not carrying any stores. In this case, the SMS will show fault codes for lost communication with both the CRIU and NRIU, which would be normal for that station's current configuration.

The logical solution to the configuration problem was to notify MDAS of the aircraft's configuration during the initialization sequence so that appropriate symptoms could be ignored and faults eliminated from the plausible set prior to diagnostics. This could easily be accomplished by creating another initialization screen similar to the screens used for initializing availability and criticality. The problem would then be reduced to what type of screen would be best to display for the technician. Three options were identified: a verbal screen (text only), a graphical screen, or a combination of text and graphics. From our human interface studies, it was apparent that strictly verbal initialization screens were despised, whereas the combination of text and graphics was well received. The option of a mixed screen of text and graphics was selected. The screen MDAS will display is shown in Figure 8.

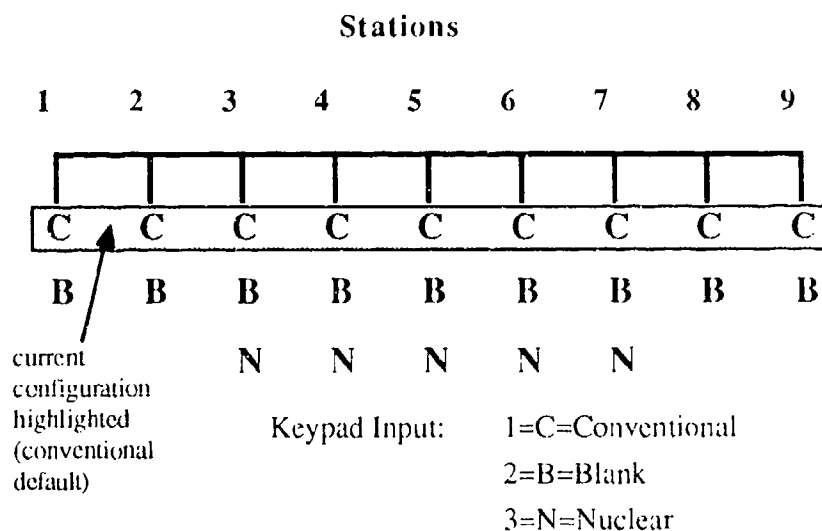


Figure 8. Aircraft Configuration Initialization Display.

The default configuration is conventional for all stations. The current configuration is highlighted, and the optional configurations for each station are shown below each station number. To set the aircraft configuration, the technician simply views this display, compares the configuration shown with that of the actual aircraft, and makes appropriate changes by selecting C, B, or N as it applies to a specific station. This configuration data will be linked to the F-16 C/D SMS diagnostic model to facilitate the exclusion of faults and flagging of symptoms which are normal for the current aircraft configuration.

### Interleaving Tests/Actions

An additional enhancement to the MDAS data display is achieved by interleaving tests and actions. The baseline version of MDAS was capable of displaying either a list of tests, a list of actions, or a list containing single (dominant) action followed by a list of tests. This is a limited capability in that the technician cannot view a mixed hierarchal list of tests and actions which lead to fault rectification. A mixed hierarchal list of the top five or ten actions provides convenient viewing of the options that will best lead to fault rectification.

The method of implementing this facility was to use the existing feedback loop in MDAS, which evaluates tests and actions, to generate a ranked list of options. The loop already performed the basic evaluating and ranking functions and, with little modification, was broadened to include this facility.

The first step after entering the loop is to initiate the multiple fault algorithms to generate a ranked list of fault sets, which represent the actions (component swaps) against which tests will be ranked. Next, using the methodology outlined for selecting a best test, MDAS performs analyses and selects best tests for the given information, ranking these tests in decreasing order. The best test list is then compared against the actions. This is done by comparing the best action and the first test in the best test list using the dominant action equation. The dominant action equation below computes times to accomplish the particular action and test and provides a decision whether to test or replace. The test or action which wins this comparison will become the first option in the list of interleaved tests/actions. It is then removed from further comparison. The test or action which loses will be compared against the next test or action in the opposite list. This process of comparison using the dominant action equation continues until the list of interleaved tests and actions has five entries, at which time the routine is terminated. For example, if ACTION A is chosen over TEST 1, the first time through, then ACTION A goes to the top of the list and is removed from consideration, and the comparison is executed again. The second time through, ACTION B is compared to TEST 1. If TEST 1 dominates, then it is placed on the list below ACTION A, the first choice; thus, TEST 1 is removed from consideration, and the loop is executed again. The third time through, TEST 2 is compared to ACTION B, and whichever dominates will be placed next on the list. A possible display of the top five options would be in the following format:

1. ACTION A
2. TEST 1
3. ACTION B
4. ACTION C
5. TEST 2

### Test Equipment Availability

In many situations, parts availability plays an important role in solution of a maintenance problem, and the baseline version of MDAS takes this factor into account. The concept of availability can be extended to include the equipment necessary to complete the diagnostics and resulting maintenance actions. The test equipment availability feature of MDAS takes into account the availability of test equipment and its direct effect on the ability to complete the recommended diagnostic tests. Consider a situation in which MDAS has selected test n as the best option, but the equipment to perform test n is presently inoperative or unavailable, making the test a less than optimal choice since it cannot be readily accomplished. MDAS should also consider test equipment availability when selecting the best option, as this would alleviate some frustration on the part of the technician faced with performing a test without the necessary equipment. This would save both time and money, since the technician will be warned against pursuing an action that cannot be performed. The test equipment availability would be set during the initialization phase of MDAS.

and the user would have the opportunity to tell MDAS if any test equipment is unavailable. If the user fails to input data on unavailable equipment, MDAS assumes all equipment is available. Once the availability is set, MDAS finds the tests which are affected, if any, and marks them for future reference. When a test is selected, MDAS checks to see if it is marked; if it is, MDAS displays the test as an invalid option. Although a test may not be a valid option, it does not affect the plausible set under investigation, and the probability of the corresponding fault or combination of faults of that test will rank the associated repair action appropriately in the interleaved list of tests and actions.

### Test Equipment Availability Implementation

This feature is fairly simple to implement because of the existing availability and criticality functions and screens; the test equipment availability function is simply a derivative of these. Application of this concept on the F-16 SMS illustrates the utility of this feature. The major effort for implementation of this feature is concentrated in two tasks: determining all the test equipment associated with diagnosing the SMS and linking these data with the appropriate tests. Research into the problem led to the conclusion that only two types of test equipment are required for testing the SMS: a multimeter and an armaments circuit preload test set. The second task was a bit more involved, as more options were available for implementation.

### Implementation Options

Two basic methods of implementing this feature were explored. The first method involves flagging and removing invalid tests immediately after availability is set. If the test equipment availability were very definite, then this method would remove "impossible" tests prior to diagnostics; MDAS would not waste time selecting a test that cannot be done. This option saves computation time but clouds good tests. The technician cannot see the test that is normally chosen, giving him a distorted view of the proper diagnostic sequence. This method seemed restricting, which is incompatible with the results of the human interface study, and was not chosen.

In the second method, after the test equipment availability is set, appropriate tests are flagged but not excluded from the diagnostic sequence. Once MDAS has selected a test, prior to its display MDAS checks to see if the test has been flagged as invalid, and, if so, displays the test as an unavailable choice in reverse video. This method allows the technician to see all tests relevant to the current diagnostic situation and to view how the equipment availability has affected the final selected options. An additional merit of this method is the ability of the technician to go ahead and perform tests previously assumed invalid. This might be applicable if the job were suspended and revisited under new availability criteria. However, this flexibility is gained at the expense of including invalid tests. This second method was chosen to provide the technician with as much viable information as possible in the event of a change in test equipment availability.

### Reinitialization/Change in Symptom

This feature enables MDAS to react to changes in the diagnostic situation by updating parameters during diagnostics. Changes in symptoms might occur if during rectification another symptom is uncovered or removed. This takes into account masked faults. As MDAS executes and the technician is applying the information to the problem at hand, certain information is gained; tests are passed or failed, and components are exculpated from the plausible set. This information is useful to MDAS as it reduces the complexity of the problem and brings the problem closer to an end.

For example, assume MDAS begins diagnostics with a set of symptoms implicating a given number of faults. As the diagnostic process proceeds, symptoms are eliminated as faults are isolated and rectified. Assume a specific symptom has been eliminated and, with it, several faults

are removed from consideration. There still remain other symptoms and faults to be removed; however, the complexity of the problem may be reduced. One of the exculpated faults might be implicated by one of the remaining symptoms. By knowing that this fault is exculpated, the plausible set of faults for the symptom being investigated is reduced, and the resulting computations are simpler and quicker. The ability to reinitialize and account for a change in symptoms is important if MDAS is to effectively attack a problem. Several options were identified concerning how such a capability might be implemented in MDAS. They are described below.

Three options were explored to possibly implement an effective reinitialization/change in symptom function in MDAS:

1. Continue. Finish the problem at hand before dealing with new information. In this method, the user operates on the initial set of symptoms until all are fixed. Then a system check is performed to determine if more symptoms are present.
2. Hard Exit. Upon removal of one symptom, the diagnostic loop is exited, at which time the control is returned to a point where new or existing symptoms are input and reinitialization occurs.
3. Immediate Update. Whenever a symptom is removed, a system check is performed, and symptoms are removed from or added to the remaining set. Data are input "on the fly," and the loop is never exited.

As these methods were evaluated, it became evident that Options 1 and 2 are inferior to Option 3. Option 1 basically ignores the numerous possibilities that can occur with the advent of a change in symptoms. By failing to promptly recognize changes in symptoms, this method has the potential to waste a lot of time. Option 2 is not a good choice because it completely wipes out the remaining symptoms prior to initialization, losing valuable information in the process. The best choice is Option 3, which simply adds or deletes information as necessary after a system check; no information is lost, and any changes in the state of the problem will be handled and incorporated in the succeeding diagnostic steps.

#### Feedback Analysis

Feedback analysis, as defined and described in this task, is the process of collecting parameters while in the maintenance/diagnostic environment and using these field values to update and mature the logistic parameters predicted in the mission phase for the equipment of interest. Parameters such as system availability, Mean Time to Repair (MTTR), Mean Time Between Failure (MTBF), and manpower or support equipment requirements are predicted during the design portion of the acquisition cycle and are based upon comparability analysis history of similar equipment. These logistic parameters are used to define levels of need for spares, manpower, and support equipment (SE). When a system becomes operational, actual values can be computed if proper parameters are collected and substituted in the original calculations for the comparability-based values.

The underlying elements used to compute the above parameters are well suited for a collection by the diagnostic routine (MDAS). The time required for each individual task within a procedure, the frequency of occurrence of each symptom within a system, the probability of success of a specific repair for a given symptom, and elapsed time to accomplish the complete maintenance action are available by selectively collecting major event information. For example, the difference between "Select Symptom" time and "Diagnostics Complete" time is the elapsed time for the completed maintenance action. Potential uses of this parameter are in the area of manpower and SE requirements.

The tool developed to demonstrate the use of MDAS collected data for maturing logistics parameters was a modification of routines developed earlier. The exhaustive look-ahead routine was designed to calculate the number of steps, diagnostic time, rectification time, and probability of occurrence for each mode, as it reduces the list of possible causes in a recursive manner. The elements collectable from running an exhaustive look-ahead for each symptom within the system being analyzed can be used to produce numerous logistics parameters. If the system is in the development stage, these parameters are predictions. If this is accomplished after production has started, these parameters may be used to update the predicted values. Figure 9 is an example of an exhaustive look-ahead output that provides all options for the repair of Symptom 9410AE of the F-16A SMS system and the associated probabilities of success. The following formula is used to calculate the Mean Time to Repair (MTTR) a given symptom based on probabilities of success and time to rectify given actions.

$$MTTR\ 9410AE = \sum \{ (Time\ to\ Rectify) \times (Probability\ of\ Success) \}$$

Figure 10 exhibits all possible exhaustive look-ahead outcomes of diagnostics for Symptom 9410AE and clarifies steps taken, paths, and corresponding probabilities of success for the exhaustive look-ahead output in Figure 9.

The other routine used was the log file. The log file collects the major events that occur when the diagnostic routine is exercised as a diagnostic aid. Information collected includes time and date of event, an event identification classification, and event results. The sample log file output in Figure 11 displays the diagnostic sequence to repair symptoms. Figure 12 explains event numbers that lead to rectifications. By accumulating these log files over time and analyzing the elements, logistics parameters can be computed which reflect real-world values. When compared to predicted values, shortfalls within the logistics area are evident and corrections can be accomplished.

### Feedback Analysis Theory

In order to generate and project requirements (spares, manpower, and support equipment), numerous logistics parameters must be collected and analyzed. MDAS can assist in collection of some of these parameters; however, since diagnostics is not the only user of spares, manpower, or support equipment, analysis conducted with only MDAS-collected parameters will be incomplete. Data associated with preventive or scheduled maintenance also need to be collected.

The distribution of time "cost" is an obvious application of the exhaustive look ahead of MDAS capability. These distributions can be compiled for each symptom, and when compared to frequency of symptom occurrence, they can be aggregated to give a mean time and cost to repair for the subsystem in which the symptom lies. Frequency of symptom occurrence or symptom weighting is defined as the amount of contribution or influence of the individual symptom when compared to the whole.

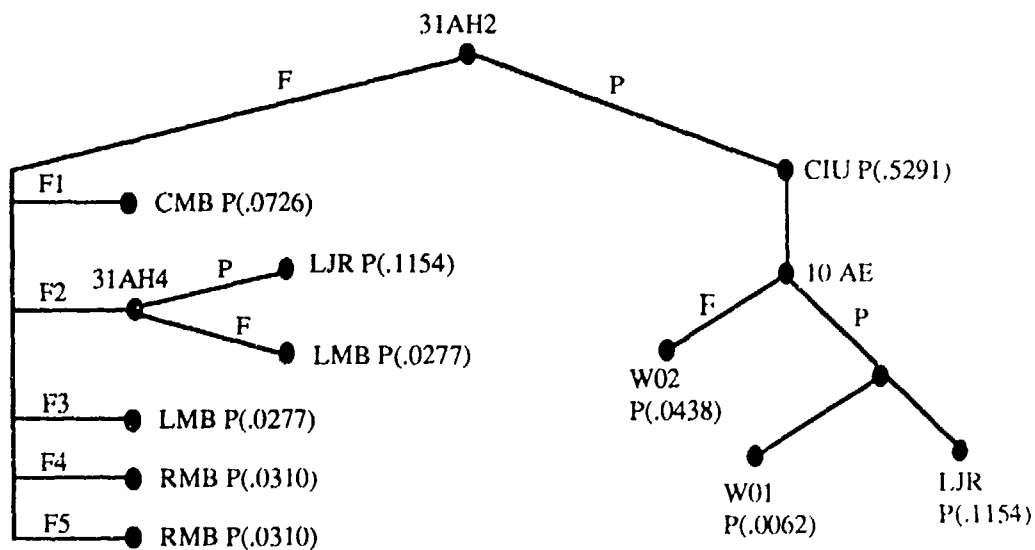
The time to repair can be computed and plotted as a mean time per symptom, minimum/maximum time per symptom, or a combination of these. Figures 13 and 14 illustrate two different display formats of a sample system. Figure 13 shows a graph of minimum, maximum, and mean projected times for Symptoms 9410AD through 9410AG associated with the F-16 SMS. Figure 14 shows the times in a table format, along with equal weighting of symptoms. The weighting of the symptoms allows individual values to be aggregated to the subsystem level to project statistics at that level. With equal weights, the subsystem values are the average values

EXHAUSTIVE LOOK-AHEAD FOR SYMPTOM 9410AE  
OF F-16A SMS SYSTEM

Test ID	Test Results	Prob. of Occur.	Rect. ID	Fault ID in Rect.	Steps Taken	Time to Isolate	Time to Rectify	Prob. of Success	MTTR per Fault
31AH2	Pass	0.694587	CIU	13	2	70	70	0.53	37.04
10AE	Pass	0.121671	LJR	2	4	170	190	0.12	21.93
			W01	30	4	170	315	0.01	1.96
10AE	Fail	0.04379	W02	4	3	100	215	0.04	9.41
31AH2	Fail 1	0.072566	CMB	33	1	10	85	0.07	6.17
31AH2	Fail 2	0.143173							
31AH4	Pass	0.115446	LJR	3	2	20	90	0.12	10.39
31AH4	Fail	0.027727	LMB	34	2	20	95	0.03	2.63
31AH2	Fail 3	0.027727	LMB	35	1	10	85	0.03	2.36
31AH2	Fail 4	0.030973	RMB	31	1	10	80	0.03	2.48
31AH2	Fail 5	0.030973	RMB	32	1	10	80	0.03	2.48
Sums							MTTR per Sym AE 96.85		

Figure 9. Sample Exhaustive Look-Ahead Output.

**SYMPTOM 9410AE OF THE F-16 SMS SYSTEM**  
**All Possible Diagnostic Paths and Associated Probabilities of Success**



**Figure 10.** Flow Schematic for the Sample Exhaustive Look-Ahead Output.

Run #	Event #	Rect./Results	Date	Time
1	8		Wed. Oct. 14, 1987	13:30
1	10	9410AG	Wed. Oct. 14, 1987	13:41
1	22	CIU	Wed. Oct. 14, 1987	13:41
1	27	0	Wed. Oct. 14, 1987	14:35
1	28		Wed. Oct. 14, 1987	14:35
2	8		Wed. Oct. 14, 1987	14:40
2	10	9410AE	Wed. Oct. 14, 1987	14:40
2	22	W02	Wed. Oct. 14, 1987	15:45
2	27	0	Wed. Oct. 14, 1987	17:20
2	28		Wed. Oct. 14, 1987	17:20
3	8		Wed. Oct. 14, 1987	14:35
3	10	9410AD	Wed. Oct. 14, 1987	14:40
3	22	CIU	Wed. Oct. 14, 1987	15:45
3	27	1	Wed. Oct. 14, 1987	17:20
3	22	RIU	Wed. Oct. 14, 1987	17:20
3	27	0	Wed. Oct. 14, 1987	18:30
3	28		Wed. Oct. 14, 1987	18:30
4	8		Wed. Oct. 14, 1987	12:10
4	10	9410AG	Wed. Oct. 14, 1987	13:41
4	22	CIU	Wed. Oct. 14, 1987	13:41
4	27	0	Wed. Oct. 14, 1987	14:35
4	28		Wed. Oct. 14, 1987	14:35
5	8		Wed. Oct. 14, 1987	14:40
5	10	9410AE	Wed. Oct. 14, 1987	14:40
5	22	W02	Wed. Oct. 14, 1987	15:45
5	27	0	Wed. Oct. 14, 1987	17:20
5	28		Wed. Oct. 14, 1987	19:35

Figure 11. Sample Log File Output.

Event Numbers	Title
8	Initialization of Diagnostics
10	Symptom Identification
22	Unit Replacement Start Time
27	Functional Check
	0 - pass
	1 - fail
28	Completion of Diagnostics

Figure 12. List of Log File Event Numbers.

based on the number of symptoms within the subsystem. Figure 15 below gives a table format with varying weights for the frequency of symptom occurrences. The subsystem values are mean values based on the individual contribution of each symptom. A difference in times can be seen when compared to the subsystem values of Figure 14.

$$\text{symptom weight} = \frac{\text{\# of times individual symptom seen}}{\text{total symptoms seen}}$$

Symptom

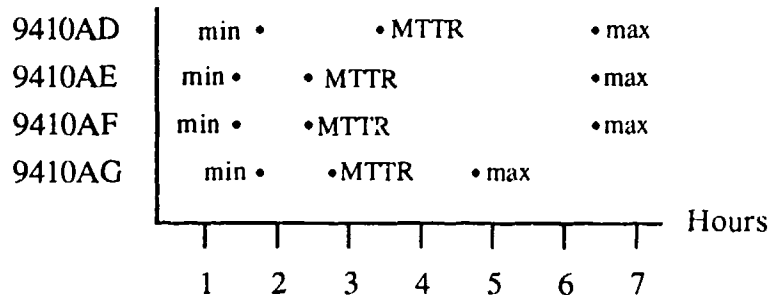


Figure 13. Feedback Analysis Example System A1.

Symptom	weights	MTTR (hours)		
		min	mean	max
9410AD	.25	2.0	3.2	6.2
9410AE	.25	1.0	2.4	6.2
9410AF	.25	1.0	2.4	6.2
9410AG	.25	1.8	2.6	4.6
subsystem	1.0	1.45	2.65	5.8

Figure 14. Feedback Analysis Example System A2.

Symptom	weights	MTTR (hours)		
		min	mean	max
9410AD	.15	2.0	3.2	6.2
9410AE	.20	1.0	2.4	6.2
9410AF	.20	1.0	2.4	6.2
9410AG	.45	1.8	2.6	4.6
subsystem	1.0	1.51	2.61	5.48

Figure 15. Feedback Analysis Example System B.

The inherent availability rate of the subsystem can be computed using information projected by the exhaustive look-ahead routine and predicted MTBF. It can then be modified and updated as actual field data become available. The actual field data are used to modify the projected values of Mean Time To Repair (MTTR), Mean Time Between Maintenance (MTBM), Mean Time Between Failure (MTBF), and task time. The MTBF is the time between component installation and failure

and is different from MTBM which is the time between any maintenance activity. The calculations made using these values will mature to the point where they become real-time values being experienced by the field.

Inherent Availability ( $A_i$ ) is the probability that, when used under stated conditions in an ideal support environment without consideration for preventive action, a system will operate satisfactorily at any time. The "ideal support environment" referred to exists when the stipulated tools, parts, skilled manpower, manuals, support equipment, and other support items required are available. Inherent availability excludes whatever preventive maintenance downtime, supply downtime, and administrative downtime may require.  $A_i$  can be expressed by the following formula:<sup>2</sup>

$$A_i = \frac{MTBF}{MTBF + MTTR}$$

where MTBF = Mean Time Between Failure

MTTR = Mean Time To Repair

The inherent availability is calculated using failure data that MDAS is capable of providing. The data require minimal manipulation to compile from individual symptoms up to the subsystem level for the parameters of MTBF and MTTR. The MTTR was calculated and used in the time to repair. The MTBF can be computed by dividing the system operating hours by the total number of system failures. The system failure rate is the inverse of the system MTBF. Operating hours are not collected by MDAS. They will, however, be available for use in feedback analysis through a Core Automated Manufacturing System (CAMS) interface which is programmed as part of the Integrated Maintenance Information System (IMIS) system. By using these values, the  $A_i$  can be calculated and projected for subsystems. By allowing the parameters to be modified and updated by field data, the  $A_i$  can mature to project real-world availability rates.

The other availability rates as defined in MIL-STD-1388-2A, DoD Requirements for a Logistic Support Analysis Record, require parameters which are not part of the diagnostics routine but may be part of the overall IMIS collection effort. Parameters such as preventive maintenance times, standby time, or mean maintenance downtime which will be required for calculating operational availability ( $A_o$ ) or achieved availability ( $A_a$ ) can be developed using the IMIS interface to CAMS.

#### Feedback Analysis Comparison

The exhaustive look-ahead routine provides data needed to forecast logistics parameters. Data such as times to accomplish repairs and the percentages of occurrence are direct outputs and require no modification for use in calculations needed to make predictions. The log file as developed in earlier work required minor modification to include a counter to individually number the simulated run to assist in maintaining identification. The output was reformatted to allow ease of entry into a data base of choice. This effort was simulated with the use of the integrated software package Symphony, on an IBM XT. Symphony provides the necessary commands to manipulate data in the data base and to perform the mathematical calculations.

<sup>2</sup>The measurement bases for MTBF and MTTR must be consistent when calculating  $A_i$ . Do not combine a time measurement base with other measurement bases such as cycles of landing gear, rounds fired, or engine starts.

External inputs which are not collected by MDAS but are gathered from within the complete IMIS process are crew size and support equipment necessary for the accomplishment of each task. Probability of symptom occurrence is used to modify MTTRs, mean manhours, and Support Equipment (SE) requirements. The number of repairs per flying hour is calculated using the probability of success from the exhaustive look-ahead for each repair as a modifier of the total number of failures projected by the failures per system flying hour.

$$(\text{Symptom MTTR}) (\text{Prob. of Symptom Occurrence}) = \text{Symptom MTTR Weight}$$

$$\sum \text{Symptom MTTR Weights} = \text{System MTTR}$$

$$(\text{Symptom Manhours}) (\text{Prob. of Symptom Occurrence}) = \text{Symptom Manhour Weight}$$

$$\sum \text{Symptom Manhour Weights} = \text{System Mean Manhours}$$

$$(\text{Symptom MTTR Weight}) (\text{Support Equip. Required}) = \text{Weighted SE Required In Hours}$$

$$\sum \text{Weighted SE Required} = \text{Mean SE Required In Hours}$$

$$\text{System Failure Rate} = \frac{1}{\text{MTBF}}$$

The actual task times and rectification choices were simulated for this effort. The collection mimics the log file output of MDAS and simulates numerous sequences of repairs. Figure 13 shows a formatted output of selected events. Analysis of these events can give symptom MTTRs and allows them to be summed in order to derive a system MTTR. Also, the number of individual symptoms and total symptoms can be counted, allowing computation of symptom probability of occurrence. This allows symptom weighting to reflect actual occurrences to weight the parameters. Other analysis can give number of repairs and Unnecessary Removals (URs) per rectification.

The summary report shown in Figure 16 allows comparison between predicted and actual values. Analysis can be made based upon these comparisons and the logistics portion of the system modified as needed. If this analysis were at the base level, then the results could be aggregated to theater and fleet levels by weighting each against the total. Figure 17 shows an aggregation flow from base level to fleet level.

#### Update Information Algorithm

This portion of the task effort compared the split-half formulation originally developed in MDAS with the "Information Algorithm," commonly known as the entropy equation, frequently used in other test selection procedures. The best test algorithm currently used in MDAS is one which seeks to maximize information gained through a test by determining which among available tests comes closest to evenly splitting the plausible fault probability distribution among spanned and unspanned faults. The MDAS algorithm is of the form:

$$TV = \sum P(S) \cdot \sum P(U) \quad (6)$$

where,

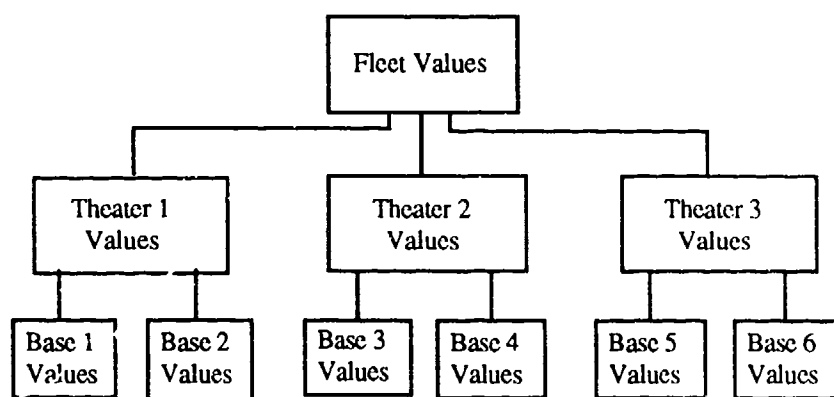
TV = Test Value,

$\sum P(S)$  = The cumulative probability of the spanned plausible faults, for a particular test.

Examples of comparisons between predicted and simulated actual parameters using the F-16 Stores Management System data as test data

Parameters		Predicted	Actual
Manpower (in hours per failure)		1.90	3.93
Spares	Rect. ID	# of Repairs (Predicted)	# of Repairs (Actual)
	CIU	7.46	8.00
	CMB	0.17	0.00
	DFS	0.12	0.00
	EJG	0.00	0.00
	EJP	0.00	0.00
	EJS	0.03	0.00
	GSA	0.04	0.00
	LJR	0.54	0.00
	LMB	0.13	0.00
	MAC	0.01	0.00
	RIU	0.18	4.00
	RJR	0.54	0.00
	RMB	0.15	0.00
	SCP	0.12	0.00
	SJC	0.01	0.00
	SIP	0.01	0.00
	W01	0.07	0.00
	W02	0.10	4.00
	W03	0.10	0.00
	W04	0.01	0.00
	W05	0.10	0.00
	W06	0.00	0.00
	W07	0.10	0.00
Support Equipment (in hours per failure)		0.89	1.60
MTTR (hours)	<u>Symptom</u>		
	9410AD	1.88	3.91
	9410AE	1.61	3.07
	9410AF	1.61	0.00
	9410AG	1.17	1.73
	System Totals	1.41	2.90
MTBF	External Input	500.00	
	Computed		416.67
Availability Rates (inherent)		0.9972	0.9931

Figure 16. Comparison Between Predicted and Simulated Actual Values.



$$\text{Fleet Values} = \sum (\text{theater values}(i) \cdot \text{theater weights})$$

$$\text{Theater Weights} = \frac{\# \text{ of occurrences at theater}}{\# \text{ of total occurrences at fleet}}$$

$$\text{Theater Values} = \sum (\text{base values}(i) \cdot \text{base weights})$$

$$\text{Base}(i) \text{ Weights} = \frac{\# \text{ of occurrences at base}}{\# \text{ of total occurrences at theater}}$$

Figure 17. Feedback Flow from Base Level to Fleet Level.

$\sum P(U)$  = The cumulative probability of the unspanned plausible faults, for a particular test.

The actual algorithm in MDAS is slightly more complicated than shown, as it also has to account for Multiple Outcome Tests, test performance time, and access group information; however, for this comparison, this abbreviated form of the algorithm is sufficient in that the modifications of the test value act solely as scaling factors.

The entropy equation is of the form:

$$TV = \sum |P_i \cdot \log_2 P_i| \quad (7)$$

where,

TV = Test Value

$P_i$  = The sum of the individual fault probabilities of the plausible faults which can lead to a failed test.

There are two significant differences between these formulations of the test value equations. Although each seeks to optimize the selection of a best test, the view of what constitutes a best test is different for the two formulations. The first and most obvious difference lies in the fact that the split-half algorithm employed in MDAS explicitly deals with the value of the faults not spanned by a test. In any single fault situation where a binary test result is expected, there is just as much information gained from a failed test as there is from a passed test.

Consequently, explicit evaluation of the unspanned plausible faults was a design goal in the original development of the MDAS test evaluation algorithm. On the other hand, the information theory algorithm deals only with the spanned faults, and unspanned faults are dealt with only by inference.

The second significant difference between the two algorithms lies in the mathematical maximum value of the formulation. Differentiating the equations with respect to the  $P(S)$  variable, setting the result equal to zero, and solving for  $P(S)$  will yield values as follows:

$$P(S)_{MDAS} = 0.500$$

$$P(S)_{Entropy} = 0.368$$

The different maxima reflect the different philosophies which governed the development of the equations. Whereas the MDAS equation was developed to explicitly examine the value of the unspanned faults in an individual test, the entropy equation seeks to "...partition the most highly suspected faults into different sets" (Towne, 1987). The MDAS equation examines available tests to find which comes closest to having the maximum likelihood faults divided between spanned and unspanned sets. The entropy equation examines available tests to find one which contains the most likely fault but excludes the next most likely. The results of these goals are essentially equal.

As there are noticeable differences between the two equations, it follows that there may be marked differences in the results. Consequently, a number of test cases were run to see what differences in "best" test selection would result. Three types of basic tests were run:

1. A binary test case of 10 faults, each separated by a test that spanned all faults which occurred "upstream" of the test (see Figure 18).
2. A series of multiple outcome tests in which each outcome spanned only one of the plausible faults but the total size of the plausible set was varied over values from 4 to 100.
3. A series of imperfect multiple outcome tests in which the number of spanned faults was always greater than 1 in each outcome.

#### Binary Case

The results of the binary test case were surprising. It might be expected that the two formulations would have virtually equal results from this sample case for which both had been designed from the beginning; however, this was not the case. The system model is shown in Figure 18, and the fault isolation trees which resulted from each of the two formulations are shown in Figures 19 and 20. Each fault in the system has fault probability 0.10 assigned.

	F	F	F	F	F	F	F	F	F	F
	1	2	3	4	5	6	7	8	9	10
T1	1	0	0	0	0	0	0	0	0	0
T2	1	1	0	0	0	0	0	0	0	0
T3	1	1	1	0	0	0	0	0	0	0
T4	1	1	1	1	0	0	0	0	0	0
T5	1	1	1	1	1	0	0	0	0	0
T6	1	1	1	1	1	1	0	0	0	0
T7	1	1	1	1	1	1	1	0	0	0
T8	1	1	1	1	1	1	1	1	0	0
T9	1	1	1	1	1	1	1	1	1	0
T10	1	1	1	1	1	1	1	1	1	1

Figure 18. Binary Test Case Model.

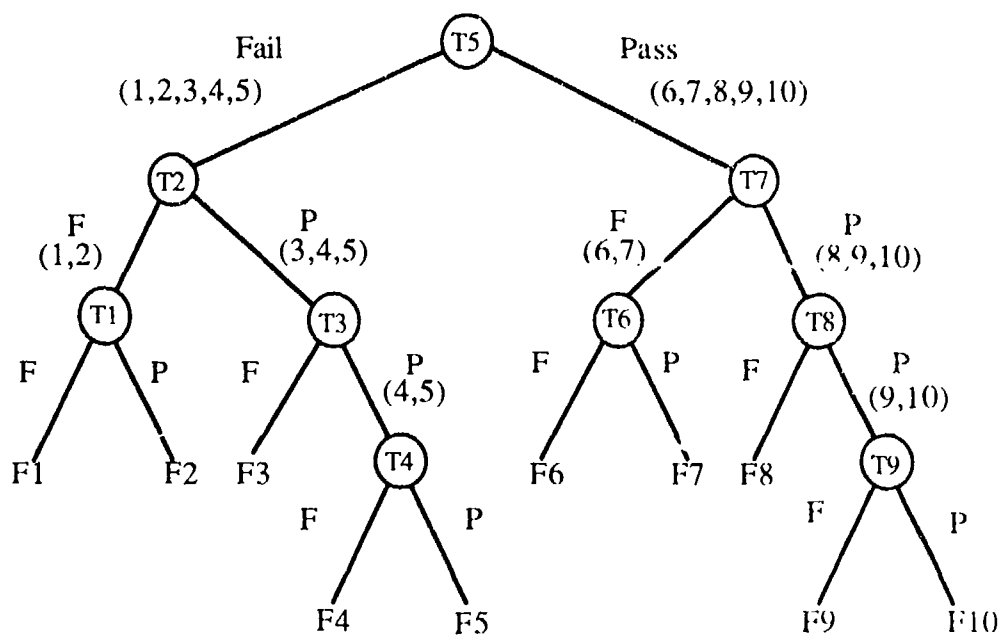


Figure 19. MDAS Formulation Isolation Tree.

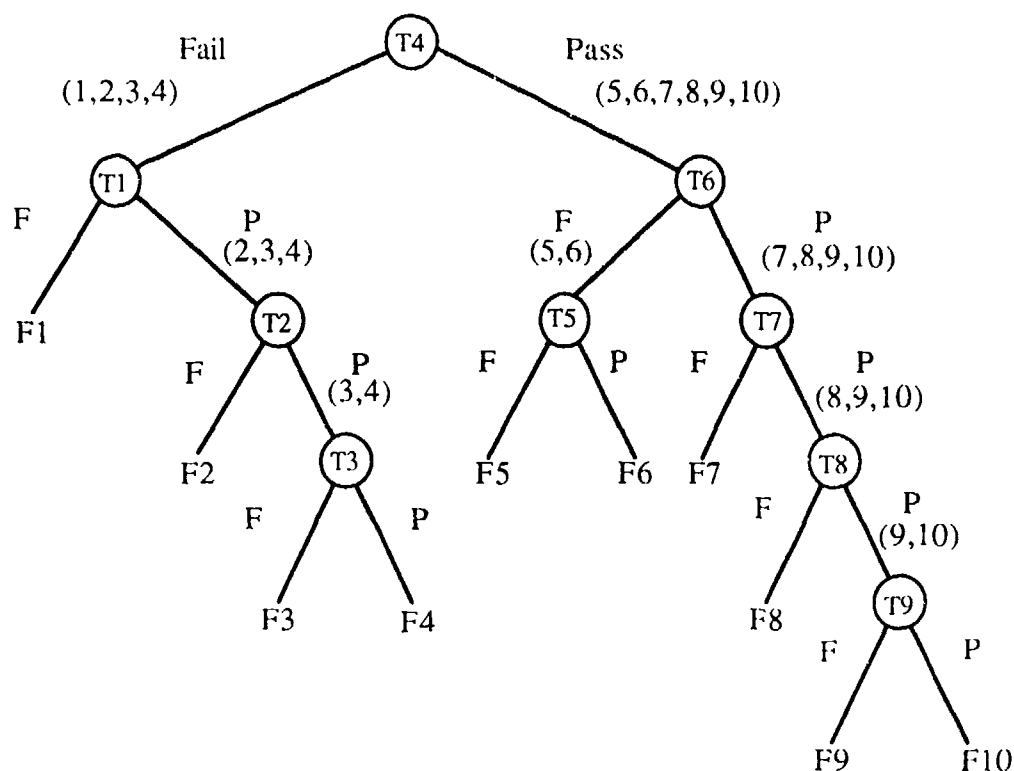


Figure 20. Entropy Formulation Fault Isolation Tree.

As one can readily see, the two fault isolation trees are quite different and their values for the average number of tests to fault isolation are also quite different. For the MDAS formulation, the average number of steps to isolation was 3.4; for the entropy formulation, the average number of steps to fault isolation was 3.6.

#### Multiple Outcome Test (MOT) Case I

The first set of multiple outcome test cases was set up such that each failed test outcome isolated the problem to a single fault in that portion of the plausible set spanned by the test. Comparison tests were run for varying plausible set sizes, with the results plotted according to how much of the plausible set was spanned by the test. Sample results are shown in Figures 21 and 22. For example, a 10-fault test was run with each succeeding calculation being a determination of test value for a continuously increasing number of outcomes until the point where there was a unique test outcome for every fault. In theory, a 10-outcome test would be a perfect test for a plausible set of 10 faults because there would be a definitive fault identification for each of the plausible faults. So long as one includes the pass line in the entropy model, results are remarkably similar to those from MDAS. Although there is a difference between the two results, no clear advantage accrues to either formulation. An example of the calculation methodology is shown with the Case II model contained in Figure 23.

## MULTIPLE OUTCOME TEST

Number of Plausible Faults = 50

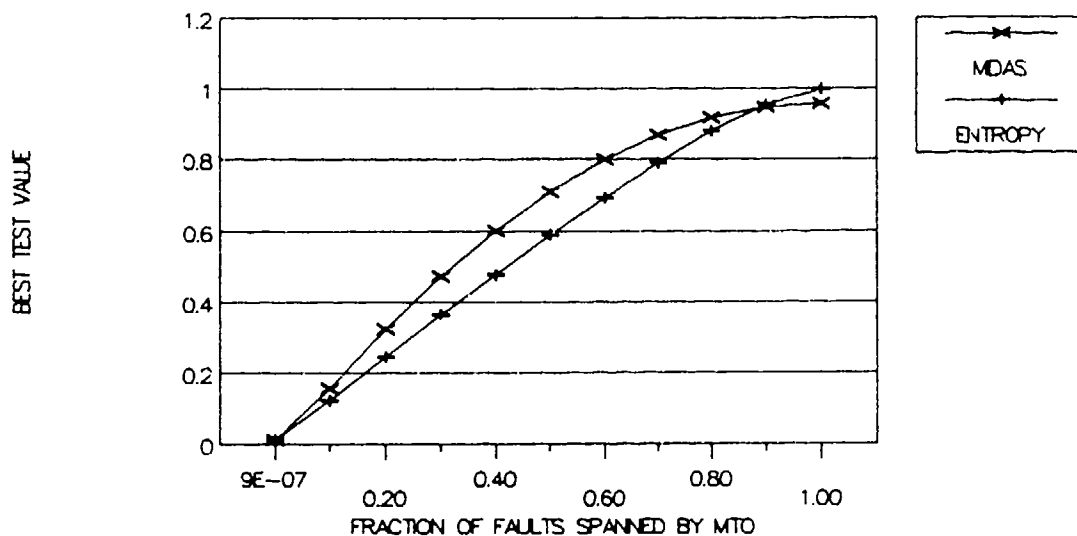


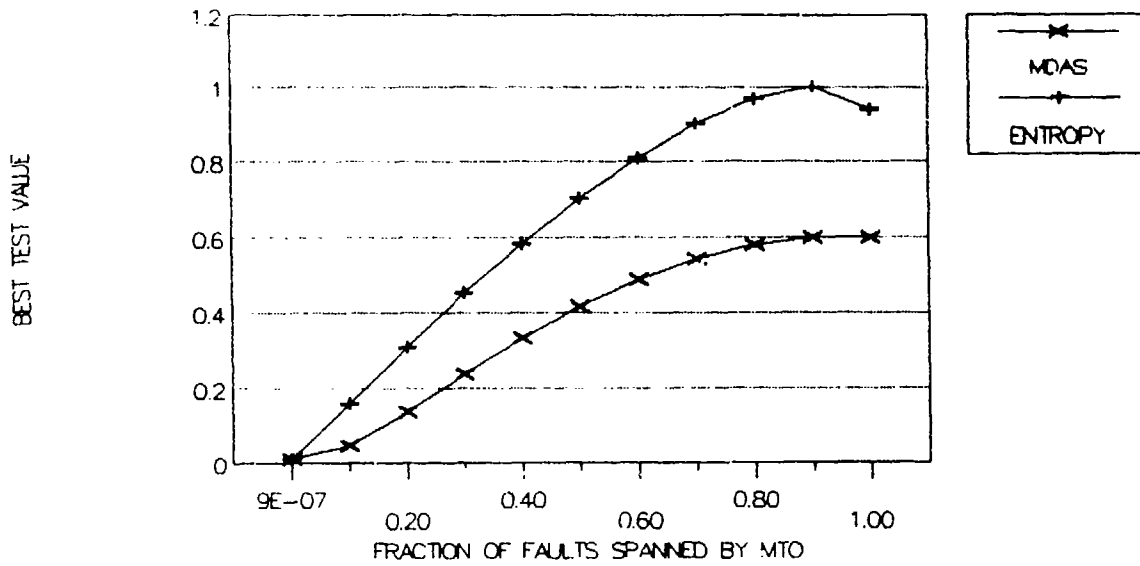
Figure 21. Case I Model, 50 Plausible Faults.

### Multiple Outcome Test Case II

A simulation to obtain test values for multiple outcome tests in which the individual test outcomes do not isolate to a single fault was run for a plausible set of eight faults for which the test spanned four of the faults in the plausible set. Test values for both the MDAS formulation and the entropy formulation were calculated for cases where each test outcome implicated three, two, and one (perfect) fault(s). The models for these three cases, along with the test values, are shown in Figures 23 through 25 below. Each faults has an equal probability  $P(.125)$  of failure, and  $R$  measures the sparseness of the MTO matrix.

## MULTIPLE OUTCOME TEST

Number of Plausible Faults = 4



## MULTIPLE OUTCOME TEST

Number of Plausible Faults = 10

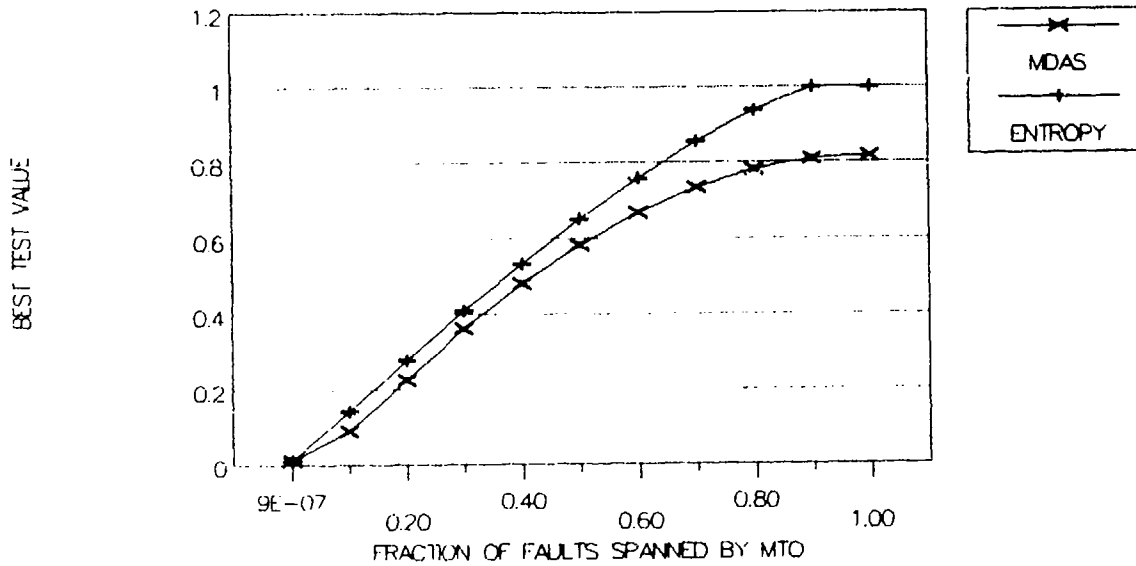


Figure 22. Case I Model, 4 and 10 Plausible Faults.

	F 1	F 2	F 3	F 4	F 5	F 6	F 7	F 8	MDAS Outcome Value	Entropy Outcome Value
Pass	0	0	0	0	1	1	1	1	(.5) (.5) = .250	.5 * log <sub>2</sub> .5 = .50
Outcome 1	1	1	1	0	0	0	0	0	(.375) (.625) = .234	.375 * log <sub>2</sub> .375 = .53
Outcome 2	1	1	0	1	0	0	0	0	(.375) (.625) = .234	.375 * log <sub>2</sub> .375 = .53
Outcome 3	1	0	1	1	0	0	0	0	(.375) (.625) = .234	.375 * log <sub>2</sub> .375 = .53
Outcome 4	0	1	1	1	0	0	0	0	(.375) (.625) = .234	.375 * log <sub>2</sub> .375 = .53
									Σ Values = 1.186	2.62

$$R = \frac{\sum 0's}{\sum 1's} = \frac{\text{sum of the \# of zero's in the table}}{\text{sum of the \# of one's in the table}} = \frac{24}{16} = 1.5$$

N = Number of Outcomes

$$\sum V = 1.186$$

$$TV = \frac{R * \sum V}{N} = 0.356 = \text{Test value}$$

2.62

Figure 23. Case II Model, 3 Implicated Faults Per Outcome.

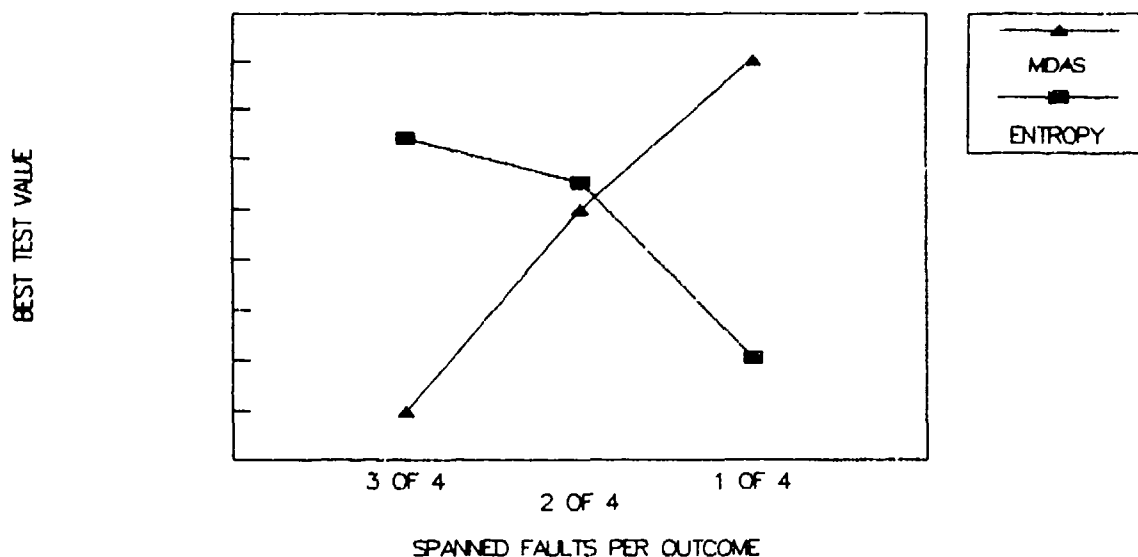
	F	F	F	F	F	F	F	F	
	1	2	3	4	5	6	7	8	
Pass	0	0	0	0	1	1	1	1	
Outcome 1	1	1	0	0	0	0	0	0	Case Test Value (MDAS) = 0.467
Outcome 2	1	0	0	1	0	0	0	0	Case Test Value (Entropy) = 2.500
Outcome 3	0	0	1	1	0	0	0	0	
Outcome 4	0	1	1	0	0	0	0	0	

Figure 24. Case II Model, 2 Implicated Faults Per Outcome.

	F	F	F	F	F	F	F	F	
	1	2	3	4	5	6	7	8	
Pass	0	0	0	0	1	1	1	1	
Outcome 1	1	0	0	0	0	0	0	0	Case Test Value (MDAS) = 0.550
Outcome 2	0	1	0	0	0	0	0	0	Case Test Value (Entropy) = 2.000
Outcome 3	0	0	1	0	0	0	0	0	
Outcome 4	0	0	1	0	0	0	0	0	

Figure 25. Case II Model, 1 Implicated Fault Per Outcome.

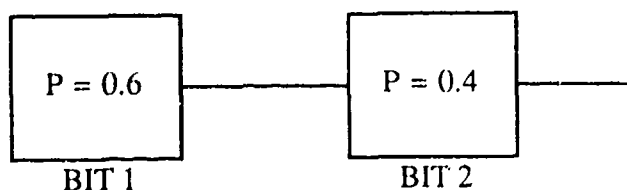
Results from this case study are shown graphically in Figure 26. The values obtained from the MDAS equation reflect the steady improvement in the actual value of the test outcomes. On the other hand, the result from the entropy equation is a line of negative slope, which indicates that the test value decreases as the precision of the test outcome increases.



**Figure 26.** MDAS vs Entropy for an Imperfect MOT.

### Different Results for Identical Values

Finally, the two equations were evaluated qualitatively for a simple system. Consider the system shown in Figure 27.



**Figure 27.** Binary Test Case.

This system can be modeled as:

	F1	F2
BIT1	1	0
BIT2	0	1

If one wishes to decide which of these two tests is the better to run at a given point in time, it is obvious that each of the tests is of equal value as the result of either test is the complement of the other. Consequently, a Test Value calculation should lead to the same result for both tests. However, while the MDAS formulation does in fact yield identical values for both tests (0.24), the entropy formulation clearly favors Test 2 (0.53 to 0.44). This result is disconcerting in that the actual physical results of the two tests are such that some other consideration such as time to perform the tests should obviously be the determining factor, but the difference in the entropy result might totally mask the advantage gained from the other consideration.

## Applicability to MDAS

Although the entropy equation used in the information theory undoubtedly is of great value in test evaluation programs, which have from the beginning been designed to handle its nuances, the disadvantage is its inability to handle multiple outcome tests as well as the current best test algorithm. We strongly recommend that the entropy equation commonly used in information theory not be considered for incorporation into MDAS.

### Wearout Failure Modes

MDAS algorithms are based on reliability theory, which assumes that faults will occur randomly in a population of components, following an established MTBF. In many situations this assumption is not valid. One situation is when all the components in question are new and enter operation simultaneously. Initially, failures in such a population will occur randomly with a more or less constant failure rate, following the established MTBF. This will continue until the total time accrued on the components of the population increases to a point where they begin to approach the end of their useful life, at which time failures will no longer be modeled by the established MTBF but will deviate significantly from this value. This deviation in failure rate ( $1/\text{MTBF}$ ) takes the form of a hump, where the failure rate increases to a maximum value then decreases back to its steady state value, and the established MTBF is again valid. Assuming that each component is replaced when it fails, subsequent humps of decreasing maxima and increasing width will result. This is a result of components being replaced at different times, making the failures of components in the population more random. Eventually, the population will become sufficiently mixed such that the humps will have dissipated to a steady state, which is well characterized by the established MTBF. This case is illustrated in Figure 28.

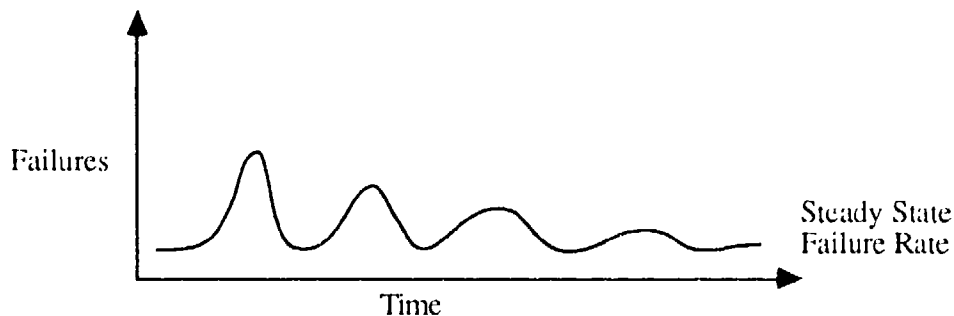


Figure 28. Failure Rate Distribution for a Population of Newly Installed Components.

If not accounted for, this phenomenon can become disastrous for diagnostic aids. Consider what would happen if MDAS were predicting the same MTBF throughout the changing failure rate cycle described above. The technician would receive recommendations based upon an MTBF that is nowhere near the instantaneous MTBF. This would not only cause confusion and extra work in rectifying the problem, but would also cause the technician to lose faith in MDAS as an effective maintenance tool. Therefore, MDAS must account for this phenomenon and make necessary adjustments made to follow the changing failure rate.

## Proposed Solutions

Numerous solutions to this problem were investigated. All of the methods attempted to enable MDAS to somehow follow the expected failure distribution for a population of newly installed components. Many of the problems encountered were related to the inability to accurately characterize this distribution. It is very difficult to generically model a function whose parameters cannot be clearly defined. The following solutions were investigated:

1. A Step Function-Like Increment of the MTBF. This method would enable MDAS to react to changes in the failure rate by incrementing or decrementing the current MTBF by a predetermined value. In doing this, MDAS would not assume a single MTBF throughout the lifetime of the component but would adjust the MTBF as necessary. The problems encountered in this approach included determining what change in failure rate was significant enough to warrant a change in the MTBF. The fact that random failures will occur prior to the hump, or "mass failure," could cause erroneous action by MDAS. Additionally, the question arose as to the value by which the MTBF should be incremented or decremented at a given time. This is a problem because failing to adjust the MTBF properly for the current situation could cause MDAS to erroneously predict too high or too low an MTBF. Due to these problems of inaccuracy, this method was abandoned. The problem of determining when the failure rate curve was departing the initial steady state sparked the following approach.

2. Goodness-of-Fit Tests. This approach involved the use of well-defined mathematical techniques to determine when there was a significant departure from the initial failure rate that would warrant an adjustment in the MTBF. The two techniques that were investigated included the Kolmogorov-Smirnov (Lilliefors, 1983; Massey, 1981) and Chi-squared goodness-of-fit tests. (Hays & Winkler, 1970). The idea behind this method was to analyze the occurrences of faults and determine if they were within the limits of some theoretical distribution and, if they were not, to update the MTBF accordingly. The Kolmogorov-Smirnov method takes a theoretical cumulative distribution function of a population and compares the cumulative step function frequency of a random sample to the specified theoretical function. This method was applied using an exponential distribution as the theoretical distribution, since such a distribution possesses a constant failure rate. One problem in applying this technique was specifying the theoretical distribution. This was a problem because the mean and variance of this distribution will never be known and will be different every time. Since the theoretical distribution was assumed to be exponential, the mean and variance were modeled by the MTBF. However, the method faltered as the number of intervals used increased -- the opposite of what was expected. The Chi-squared technique was attempted using an exponential distribution as the expected distribution. The Chi-squared distribution is a theoretical probability distribution by which adjustments to the degrees of freedom provide a skew that exemplifies a population's normal distribution about a functional  $f(x)$  mean. Comparisons of random samples can be made with respect to the theoretical probability. The results were far from useful, probably because of the limited number of samples used in the calculations. Like the Kolmogorov-Smirnov method, this technique is not useful once the failure rate curve enters the hump.

3. Computing an Instantaneous Failure Rate. In analyzing the failure rate curve, it is clear that any point along the curve represents the instantaneous failure rate. Therefore, by constructing such a curve as failures occurred, an instantaneous failure rate could be computed and the MTBF adjusted according to this new value. The problem with calculating an instantaneous failure rate is that the number of failures per unit time might have to be calculated for an extremely small unit of time. This becomes a problem because in such a small interval, a failure might not occur, and the resultant failure rate would be zero for that interval. This method seemed promising if the problem of avoiding a zero failure rate could be avoided. The solution detailed below was chosen because it avoids computing a faulty zero failure rate.

4. Sliding Window for Estimation of Failure Rate. This method was initially developed to avoid computing a fault zero failure rate due to using small intervals in which no failures occurred. The idea behind this method is to compute a failure rate based on the number of failures in a predetermined interval that contains at least one failure. The interval chosen for the calculation also had to be variable and applicable to the component under analysis. The component MTBF was chosen at the interval in which to count failures because it varies from component to component and should have at least one failure per interval. To further increase the accuracy and smoothness of the resultant failure rate curve, this interval of MTBF hours was to be moved or "slid" at increments of 20% of the MTBF, calculating a new failure rate after each increment. For example, if the MTBF was 500 hours, intervals of 500 hours would be used and the upper limit of each new interval would be incremented by 100 hours (20% of 500). The first failure rate would be computed by counting the number of failures in the interval from 500-0 hours, the second failure rate from 600-100, and so on. This method avoids the problem of computing a zero failure rate, is independent of the failure rate distribution, and is applicable for all points along the failure rate curve.

#### Development of Solution

Once the mechanics of this "sliding window" technique were more clearly defined, the method was tested manually by arbitrarily selecting an MTBF, generating a list of times at which failures occurred, generating the numerous intervals in which to count failures (MTBF in duration), and counting the number of failures in each interval. These results were then compared to the actual failure rate determined by the difference in the MTBF between the failures. The algorithm did a good job of modeling the actual failure rate curve; so, the next step in the development was to verify the method in software. To do this, the algorithm was incorporated into a BASIC program. The program was written in Microsoft Quick Basic 4.0 on an IBM XT. It was designed to handle the same data as are available to MDAS and, from those data, to generate failure rates to model the actual failure rate curve. This would demonstrate the ability of MDAS to handle simple inputs, analyze the particular situation, and adjust the MDAS data base so that the actual failure rate curve would be followed.

The program was designed to evaluate an observed failure history and make numerous computations employing the wearout failure algorithm developed for MDAS. The following information is required to execute this program:

- failure history: an ordered set of failures with corresponding times of occurrence given in total hours on all components.
- MTBF expected for the component.

From this information, the program computes:

- the "actual" failure rate. This rate is determined by counting all failures within MTBF (expected) hours prior to the most recent failure. These values are eventually plotted as the actual failure rate curve; however, this is only an approximation for the purpose of comparison with the MDAS results.
- numerous intervals. This feature provides periods in which to count failures. The intervals are MTBF (expected) hours in duration. The first interval has an upper limit of MTBF (expected) hours and subsequent intervals have an upper limit that is incremented by 20% of the expected MTBF each time.
- MTBF (calculated) for each interval. This value is derived by counting the number of failures in each of the computed intervals.

- two sets of Cartesian coordinates: one set for plotting a graph of the actual failure rate and the other to plot the calculated failure rate. The coordinates are the form (interval upper limit, number of failures in that interval).

Program output includes the following:

- a listing of times of occurrences of failures (in total hours on all components).
- a listing of calculated intervals and the number of failures in each.
- a plot of the actual failure rate versus the calculated failure rate (time versus failures).
- an output file containing the data points to reconstruct the failure rate curves in another application.

The computer simulation was run for numerous cases, which included varying the number of data points, expected MTBF, and type of failure information. The algorithm did an excellent job of modeling the actual failure rate, especially when the failure rate changed rapidly. This result was well received because this is the most critical part of the curve for MDAS. The beginning and ending of the curve were not as close to the actual value, but the deviation was never more than one failure per MTBF.

A plot of the actual and MDAS-calculated failure rates was also generated with a spreadsheet analysis program as well as by the BASIC program. Several graphs generated in the spreadsheet analysis program are included in the Appendix. These graphs illustrate the ability of the new wearout algorithm to accurately model the problem of wearout failure in a population of newly installed components.

### Wearout Failure Modes Summary

This algorithm is well suited for quickly adjusting the MTBF in the diagnostics data base. One of the reasons this algorithm is so good is that it is at its best when there are significant changes in the failure rate. When the changes are not very significant, the computed value can oscillate between two levels of failure rate. However, the fact that the goal of adjusting the MTBF is to facilitate moving a specific option to a higher priority in a list of options makes a drawback less detracting -- detailed accuracy is not necessary. This algorithm will enable rapid adjustments to the MTBF being used in the diagnostic data base, which will, in turn, provide timely information for the technician working with components experiencing wearout failure.

### Future Implementation

In order for such a function to be implemented, it would be necessary to generate failure history concerning the population under analysis. This information cannot be compiled within MDAS because failures will occur at different times and locations (different aircraft). Therefore, these failure data must be compiled at some other central location. An ideal location would be within the base-level information system. This would work well for several reasons. The base-level information system will eventually interface with the IMIS workstation for exchange of pertinent maintenance data. Failure data could be included in this exchange. Data will vary from base to base, which makes the base-level a good choice; local variation would be accounted for. This would allow for the tracking of failure history in a convenient and efficient manner. Each

base-level information system could store aircraft failure data which would include items that failed, the times of failures, etc. These data would also be manipulated in such a manner as to produce the appropriate information needed by MDAS to accomplish the wearout failure analysis.

MDAS could be used as a vehicle to update the base-level data base at the end of each day, continually updating the data base and, in turn, providing MDAS with the most recent information about the population under consideration. This could be accomplished by downloading failure data collected within MDAS via the Portable Computer Maintenance Aiding System (PCMAS). The PCMAS could then be interfaced with the IMIS workstation and base-level information system for compilation of the failure information for that day. This information could then be downloaded into MDAS prior to the following day.

The fact that MDAS cannot alone compile the information necessary to complete the wearout analysis leads to the conclusion that this feature cannot be implemented within MDAS at this time. It will have to be included in the development of the IMIS workstation, which will interface with the base-level information system to record and generate the appropriate data for use by MDAS.

### Model Repair and System Verification

This task required that options be evaluated for ways to model system verification as a set of independent steps rather than a monolithic test structure. Currently, system verification routines are not all designed to easily accommodate the verification of a single repair action. Many are essentially system functional checks which may have numerous steps. If these types of checks are used to validate simple repairs, time will waste checking items which have not been altered. The solution to this problem is to modify the system verifications to handle such circumstances. This concept was initiated as a result of the shortcomings of the F-16 SMS confidence checkout, and the resultant models are derived with the F-16 SMS, but they are intended to address confidence checkouts as a whole. The obvious benefits of this concept are to reduce the time required to verify that a repaired system is healthy, as well as to add to the flexibility of repair validations.

The confidence checkout for the F-16 SMS was evaluated using the Job Guide for the Weapons Release and Management System for the F-16 and the Fault Isolation Manual for the F-16 Weapon System. Currently, the confidence checkout for the F-16 SMS consists of a set of 221 steps which must be executed sequentially. This particular confidence checkout takes about 30 minutes to complete and is designed to be terminated when an unexpected result occurs (when a fault is found). This system verification allows the technician to verify the overall health of the system with which he or she is working.

Analysis of the F-16 SMS confidence checkout revealed that it might be modeled differently than it is presently, reducing the time required for system verification. The confidence checkout can be grouped functionally into three test groups: switch, option, and comm tests, each of which is preceded by common initialization steps. The individual steps which are included to verify each of the functions within the SMS are completed by manipulating various switches and buttons within the F-16 cockpit. The positioning of these switches and buttons is the same at the beginning of the testing for all three functional groups. Therefore, once the generic initialization steps are accomplished, any of the steps of the three functional groups can be initiated. This allows for some degree of "modularization" within the system verification. For example, if a fault is known to lie within a specific module of the confidence checkout, only that specific module will be executed when the fault is rectified. If this module passes, the remaining two modules can be executed to check the overall system health; if it fails, then diagnostics can be initiated. This will result in a substantial time savings because approximately two-thirds of the confidence checkout will have been eliminated in the initial verification of the rectified fault.

The above-described routine seems attractive but is not without uncertainty. Without detailed knowledge of the design of the F-16 SMS confidence checkout (including dependencies between individual tests and system failures), one cannot be absolutely positive that a passed test in one module of tests precludes faults appearing in remaining modules. This is the reason that the entire confidence checkout must be completed after verification of a rectified fault. Immediate implementation of such a feature within MDAS does not seem feasible until the inner details of the checkout are clearly understood.

The confidence checkout could be used differently to verify that system faults are rectified. Assume the confidence checkout is executed to determine system health, and a single fault is uncovered by a specific test. Once the fault is diagnosed and rectified, the method of validation of the repair could be modified. Rather than executing the entire set of tests or even a subset of the tests, the test which initially implicated the fault could be rerun to determine if the fault had indeed been rectified. This would alleviate numerous tests for which the probability of failing is small, thus saving time. However, in order to be confident about the system's health, the entire confidence checkout must be completed after the fault rectification is verified. This method again depends upon the detailed operation of the SMS confidence checkout. If such a method is possible, it would certainly save time.

From this analysis several recommendations may be made. First, it should be noted that modularization of system verification tests would alleviate testing components already known to be good and save time. Using the F-16 SMS confidence checkout as an example, confidence checkout modularization needs to be an engineering design consideration. Without knowing the intricate details of the SMS checkout, one cannot assume that it can be broken clearly into three sets as stated above. It would be desirable if the checkout could be broken down into functional groups which could be assessed in any order once the initialization steps had been accomplished. In order for this to happen, the test must be designed this way from the beginning. Coordination with the designers of new systems would enhance the probability of modularization.

An alternative method of modifying the confidence checkout will contribute to the multiple fault capability. Currently the confidence checkout is designed to "kick out at first failure." This means that the confidence check is terminated at the occurrence of the first fault, and that fault is diagnosed. Once the fault is rectified, the confidence check is rerun from the beginning. If this checkout were modified to continue until the tests are completed, all faults gathered in the course of the checkout can be collected and used as the initial set for multiple fault analysis. The problem with such a modification is that the exact inner workings of the SMS confidence checkout are not known; therefore, one cannot be assured that the tests following a failed test will function properly. Further investigation is needed to determine if it is a valid approach.

### User Dialogues

User Dialogues include a number of user facilities intended to further aid the technician in performing maintenance. These functions were researched and developed to comply with the human interface studies which concluded that the technician wants as much information and flexibility as possible. The following dialogue features were developed to a point where they are ready for incorporation, and recommended for the inclusion in the next version of MDAS.

#### Erase a Test

In some instances, the technician performing diagnostics may want to eliminate a previously entered test from the overall diagnostic calculations. This might occur for several reasons. The technician may have received bad information, or made a mistake in the input. If such inputs are not corrected, the output of the diagnostics will be incorrect. Additionally, the technician may wish to see what would have happened if he or she had not entered a specific test.

This has obvious training benefits in that a student can see the effect of tests on the diagnostic sequence. Aside from the training benefits, a function which will erase a previously entered test and recompute the resulting diagnostic sequence will be a valuable asset to the tool. This type of function will enhance flexibility of operation and contribute to a user-friendly device.

This function allows the technician to erase previous tests from consideration at any point in the isolation sequence. In doing this, MDAS recalculates the status of the plausible set according to the tests which were removed and also all subsequent activity, and displays the new diagnostic situation.

### Show a Symptom's Tests

Human interface studies have proven that the technician wants as much information and flexibility as possible from the diagnostic aid. A good place to implement this is in the display of diagnostic information to the technician. One useful display would be the display of all tests which give information to isolate the fault causing the symptom. This type of information would be useful because it would enable the technician to see not only which tests are available but, also how they affect the diagnostic sequence. This type of function will give technicians more information, allowing them to be more aware of the overall diagnostic situation, as well as adding flexibility to select tests.

A facility would allow the technician to view all the tests associated with a particular symptom. This feature will be accessed via a function key and display list of tests associated with a specific symptom. This list will allow the technician to select any of the tests displayed.

### Show All Tests/Actions

Another function that would enhance flexibility of operation and give the user more information would be one which shows the user all tests/actions which have been accomplished already in diagnostics. This type of function will give the technician information as to what has been accomplished, and should make for a more efficient diagnostic sequence by avoiding repeated actions. Additionally, this feature will give the technician a summary of actions which will aid in determining where the diagnostics have been and are going.

A facility would allow the technician to view all the tests or actions already accomplished in the diagnostic sequence. This feature should be accessed via a function key and, when called, produce a complete ordered list of completed tests and actions along with the result of that activity.

### Development Hardware

MDAS was developed and tested on a Sun Microsystems Model 2/170 workstation provided by AFHRL. The Sun workstation uses a MC68010 processor employing 32-bit words, a 10 MHz clock, 4 MB RAM, and one 162 MB hard disk. All user input interfaces were designed to be accomplished using only the F1 through F3 function keys and the numeric keypad to ensure commonality with the Portable Computer-Based Maintenance Aiding System (PCMAS) currently under development by AFHRL. The PCMAS will be the hardware hosting MDAS which the technician will take to the plane and use to input symptoms and diagnose the system.

### III. SOFTWARE TESTING

#### Subroutine Testing

The software developed as part of the present effort was subjected to a complete software test program upon its completion. Testing was accomplished in accordance with an approved software test plan submitted separately using both dummy data and the target data, which was the F-16 SMS data base. The software tests were designed to test each of the new functions and followed the general requirements listed below:

1. Each function was tested using nominal, extreme, and erroneous input values.
2. Each function was tested for error detection and proper recovery, including appropriate error messages.

#### System Integration Testing

Upon completion of testing each individual function, system integration was performed by linking and executing each of the accepted functions. As each function was linked, all previous integration tests were rerun to ensure the linking process had not altered the proper operation of the system and correctness of output. All integration testing conformed to the following requirements.

1. Each aggregate of integrated functions was tested using nominal, extreme, and erroneous input values.
2. Each aggregate of integrated functions was tested for error detection and proper recovery, including appropriate error messages.

Any errors detected during the testing were corrected, and the entire test sequence, for corrected algorithms, was performed again to assure compliance with the expected outcome.

### IV. DISCUSSIONS AND CONCLUSIONS

This research effort provided numerous improvements to the baseline version of MDAS. Data handling techniques were developed which deleted the need for two of the three original assumptions necessary for the baseline version, making this version of MDAS more realistic and efficient in handling the wide range of situations that can arise in the real maintenance world. A multiple fault handling strategy is given which mathematically proves that the intersection of a single fault with a set of exhibited symptoms is generally the best point at which to begin diagnostics. This method takes into account the prior probabilities of the faults being considered as well as the time required to complete various maintenance actions.

Studies and analyses of human interface issues have equipped MDAS with interface options that are likely to be effective when applied to a variety of systems and users. Sample users were unanimously in favor of displaying all actions, even actions not feasible or available; displaying a message defining the available tests and percentage of probability; displaying information with both verbal and graphical explanations; allowing users control of as much information as possible and an initialization menu to provide this control; and using function key identifiers to identify function selections.

Implementation in MDAS of the new functions and capabilities listed below will increase its utility as a diagnostic tool and human-computer interface device:

1. Suspend Function
2. Log File
3. Review Previous Action
4. Erase a Test
5. Show a Symptom's Tests
6. Show All Tests/Actions
7. Interleaving Tests/Actions

The following conclusions summarize results from this and prior related research and development efforts:

1. Cannibalization Modeling. Investigation proved that cannibalization models can be developed and implemented to aid in aircraft diagnostics and repair. Further input and output (I/O) interface studies are required to identify user input parameters, and IMIS technical and data support.
2. Reinitialization/Change in Symptoms. The incorporation of user input menus to identify symptom changes within the diagnostic loop provides MDAS with a recursive network that is reinitialized at the start of each diagnostic sequence iteration. Any changes in the state of the fault/symptom matrix are updated by user inputs and integrated into the succeeding diagnostic steps to eliminate redundant iterations and information loss.
3. Feedback Analysis. As developed, the exhaustive look-ahead routine calculates the number of diagnostic steps -- diagnostic time, rectification time, and probability of occurrences for each node. Elements collected in the Log File can be used to produce numerous logistics parameters and update predicted parameter values.
4. Update Information Algorithm. Both entropy and split-half formulations evaluate test/fault matrices for best tests. Computer simulation analyses provided no conclusive evidence to select one method over the other for most test scenarios. A difference was noted when considering best tests in multiple outcome test situations; the split-half formulation provided the better solution in these situations.
5. Wearout Failure Modes. A "Sliding Window" technique was developed to compensate for wearout failure modes that affect analysis of Mean Time Between Failure (MTBF). Computer simulations graphically verified the modeling of wearout failure modes and it is recommended that this technique be implemented into MDAS to adjust for component wearout.
6. Model Repair and System Verification. Investigation of system verification procedures determined that desirable confidence checkout tests must be adjusted to meet modularization designs of aircraft. The verification process should be designed in conjunction with functional groups, thereby allowing direct analysis of groups and reducing functional checkout time.

## REFERENCES

- Cooke, G.R., Jernigan, J.H., & Treadway, J. (1986, November). Development of a Model Based Diagnostic Aiding System (MDAS) (Contract No. F33615-85-C-0010). Logistics and Human Factors Division, Air Force Human Resources Laboratory.
- Hays, W.L., & Winkler, R.L. (1970). Statistics: Probability, inference, and decision. New York: Holt, Rinehart and Winston, Incorporated.
- Lillicfors, H.W. (1983, March). On the Kolmogorov-Smirnov Test for the exponential distribution with mean unknown. Washington D.C.: George Washington University, American Statistical Association Journal.
- Massey, F.J. (1981, March). The Kolmogorov-Smirnov Test for goodness of fit. Eugene, OR: University of Oregon, American Statistical Association Journal.
- Towne, D.M. (1987). A generic expert diagnostician. Los Angeles, CA: University of Southern California.

## BIBLIOGRAPHY

United States Air Force (1984, October). Electronic Reliability Design Handbook (MIL-HDBK-338, Vol. I). Washington, D.C.: Department of the Air Force

TG	16C-01	List of Applicable Publications
	-06	Work Unit Code Manual
	-2-00FR-00-1	Fault Reporting Manual
	-2-00GV-00-1	General Vehicle Description
	-2-00GV-00-2	General Vehicle Description
	-2-00GV-00-3	General Vehicle Description
	-2-00JG-00-1	Job Guide Index
	-2-10JG-00-1	Aircraft Safety Job Guide
	-2-94FI-00-1	Fault Isolation - Weapons Systems
	-2-94GS-00-1	General System - (Title same as 1F-16C-2-94FI-00-1)
	-2-94JG-00-1	Job Guide - (Title same as 1F-16C-2-94FI-00-1)
	-2-94JG-00-2	Job Guide - (Title same as 1F-16C-2-94FI-00-1)
	-2-94JG-10-1	Job Guide - Weapons Release and Management System, F - 16C/D Aircraft
	-2-94JG-10-2	Job Guide - (Title same as 1F-16C-2-94JG-10-1)
	-2-94JG-30-1	Job Guide - Weapons Suspension, F - 16C/D Aircraft
	-2-94JG-30-2	Job Guide - (Title same as 1F-16C-2-94JG-30-1)
	-2-94JG-30-3	Job Guide - (Title same as 1F-16C-2-94JG-30-1)
	-2-94JG-30-4	Job Guide - (Title same as 1F-16C-2-94JG-30-1)
	-4-1	Illustrated Parts Breakdown - Introduction
	-4-2	Illustrated Parts Breakdown - Numerical Index
	-4-3	Illustrated Parts Breakdown - Reference Designation Index
	-4-94	Illustrated Parts Breakdown - Weapons Systems
	-33-1-1	Nonnuclear Munitions - Basic Information

## LIST OF ABBREVIATIONS

Aa	-	Achieved Availability
Ai	-	Inherent Availability
Ao	-	Operational Availability
AFHRL	-	Air Force Human Resources Laboratory
CAMS	-	Core Automated Maintenance System
CRIU	-	Conventional Remote Interface Unit
ETIC	-	Estimated Time In Commission
HP	-	Hewlett Packard
IMIS	-	Integrated Maintenance Information System
LRU	-	Line Replaceable Unit
MB	-	Mega Bit
MDAS	-	Maintenance Diagnostic Aiding System
MTBF	-	Mean Time Between Failure
MTBM	-	Mean Time Between Maintenance
MTO	-	Multiple Test Outcome
MTTR	-	Mean Time To Repair
NRIU	-	Nuclear Remote Interface Unit
PCMAS	-	Portable Computer-Based Maintenance Aiding System
RIU	-	Remote Interface Unit
SE	-	Support Equipment
SMS	-	Stores Management System
SRU	-	Shop Replaceable Unit
UR	-	Unnecessary Removal

## GLOSSARY

Action. A diagnostic or corrective procedure performed by a maintenance technician.

Aircraft Configuration. Placements or layouts of aircraft system components.

Availability. A measure of a component's being obtainable to use in the diagnostics process.

Best Action. A multiple fault algorithm which chooses the optimum action from among available rectification actions.

Best Test. A multiple algorithm which chooses the optimum test from among those available at any point in the diagnostic sequence.

Component. The lowest physical level of indenture which a maintenance technician at a given level of maintenance organizational, intermediate, and depot (O, I, D) will normally work on. For example, an organizational level maintenance technician would consider a Line Replaceable Unit (LRU) as a component; an intermediate level technician would consider the LRU an end item and the Shop Replaceable Unit (SRU) a component.

Criticality. A measure of need for a particular system capability. For example, a fault in an air-to-ground function might not be critical for an air defense sortie, whereas a fault in an air-to-air function would be critical for the same sortie requirement.

Dominant Action. A rectification action whose likelihood of success is so great that it is recommended prior to available tests that would reduce the plausible set.

Estimated Time In Commission (ETIC). Probability that system repair will be completed in a specific amount of time.

Exhaustive Look Ahead. This method calculates the number of steps, diagnostic time, rectification time and probability of occurrences for each path of isolation that leads to a possible rectification.

Failure Rate. The inverse of Mean Time Between Failures.

Fault. The cause of an equipment malfunction. The manifestation, through either inference or direct observation, of a failure within a system.

Feedback Analysis. The process of collecting parameters while in the maintenance/diagnostic environment and using these parameters to update current logistics information.

Feedback Loop. An interconnection of faults and signals such that no single test point can successfully isolate the fault location.

Functional Check. A test performed to ensure that a rectification action has been successful in restoring a system to operational status.

Log File. A file that collects major events during diagnostics.

Look-Ahead. A computer simulation action by which the consequences from assumed results from diagnostic actions can be examined. For example, a technician might wish to know

before conducting a test what the consequences would be if it passed or if it failed. The look-ahead simulation provides this capability.

Mean Time Between Failures (MTBF). The unit of reliability used in this program as a predictor of fault likelihood. Its inverse is the failure rate.

Multiple Faults. An event where two or more faults (failed components) occur simultaneously.

Multiple Outcome Test (MOT). A test procedure which does not have a binary pass/fail result. The procedure may have any number of outcomes; however, each outcome is unique and distinguishable from all other outcomes.

Multiple Test Outcome (MTO). The unique result from a multiple outcome test.

Plausible Set. The set of possible faults which could logically be expected to have led to an observed or indicated faulty condition. The elements in this set of faults contain single faults or combinations of faults that are not redundant.

Rectification. The repair of a fault(s) which alleviates a symptom or set of symptoms.

Repair Time. The time required to complete system repair after a fault is isolated. It may include access times. It will include reinstallation of original components removed unnecessarily as part of diagnostics, secure and closure, and final functional check.

Second-Step Look Ahead. Provides a probabilistic foresight of selected actions and time required to accomplish those actions.

Support Equipment. Tools or devices needed to perform an action.

Symptom. A verbal description of the indications that a malfunction exists; e.g., "Receiver, no audio."

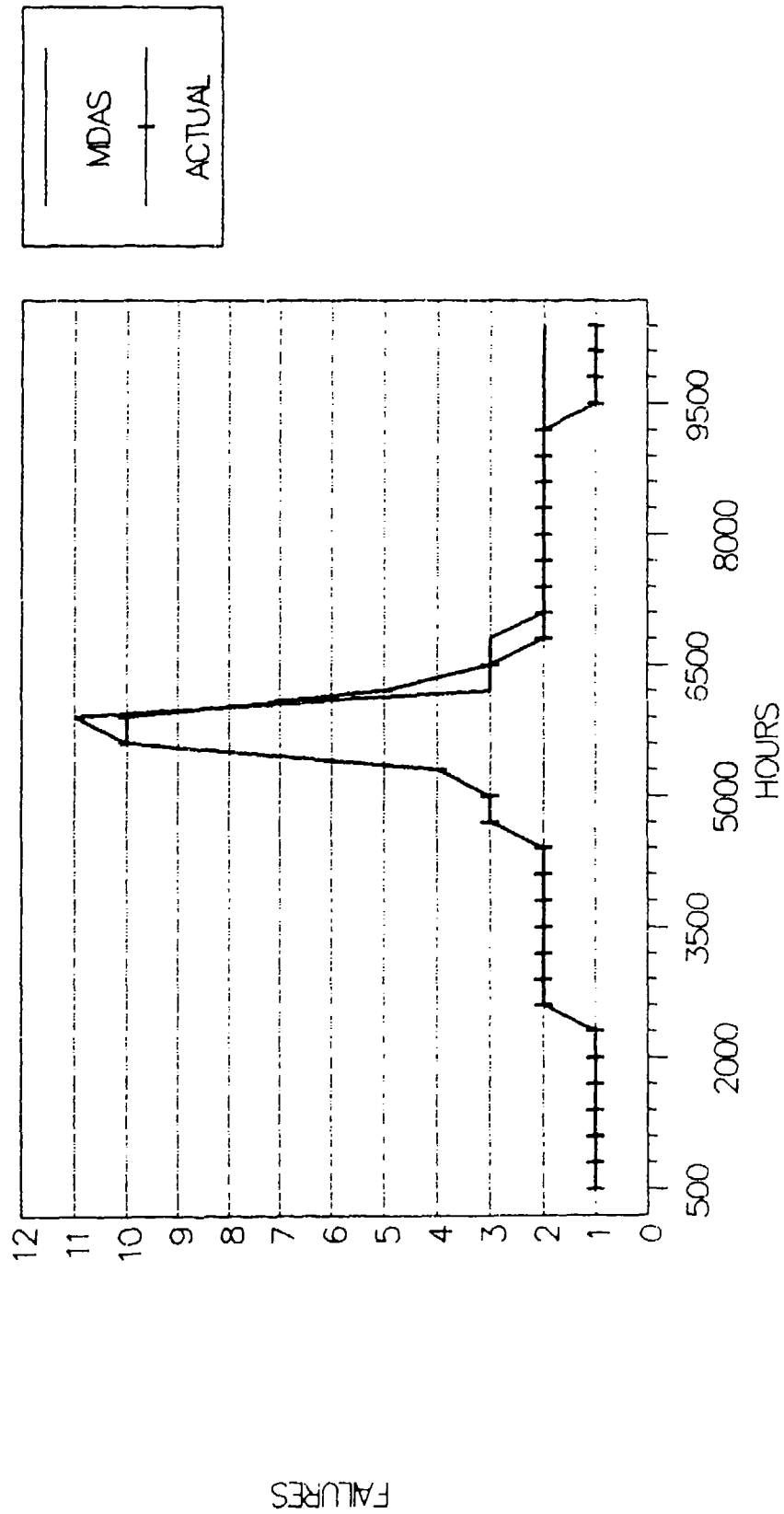
Test. A prescribed sequence of actions whose result will implicate or exonerate a set of faults.

Test Time. The time required to perform a test. It includes access time, time to gather necessary test equipment and tools, time to conduct the test procedures, and time needed to record/interpret test results.

## APPENDIX: WEAROUT GRAPHS

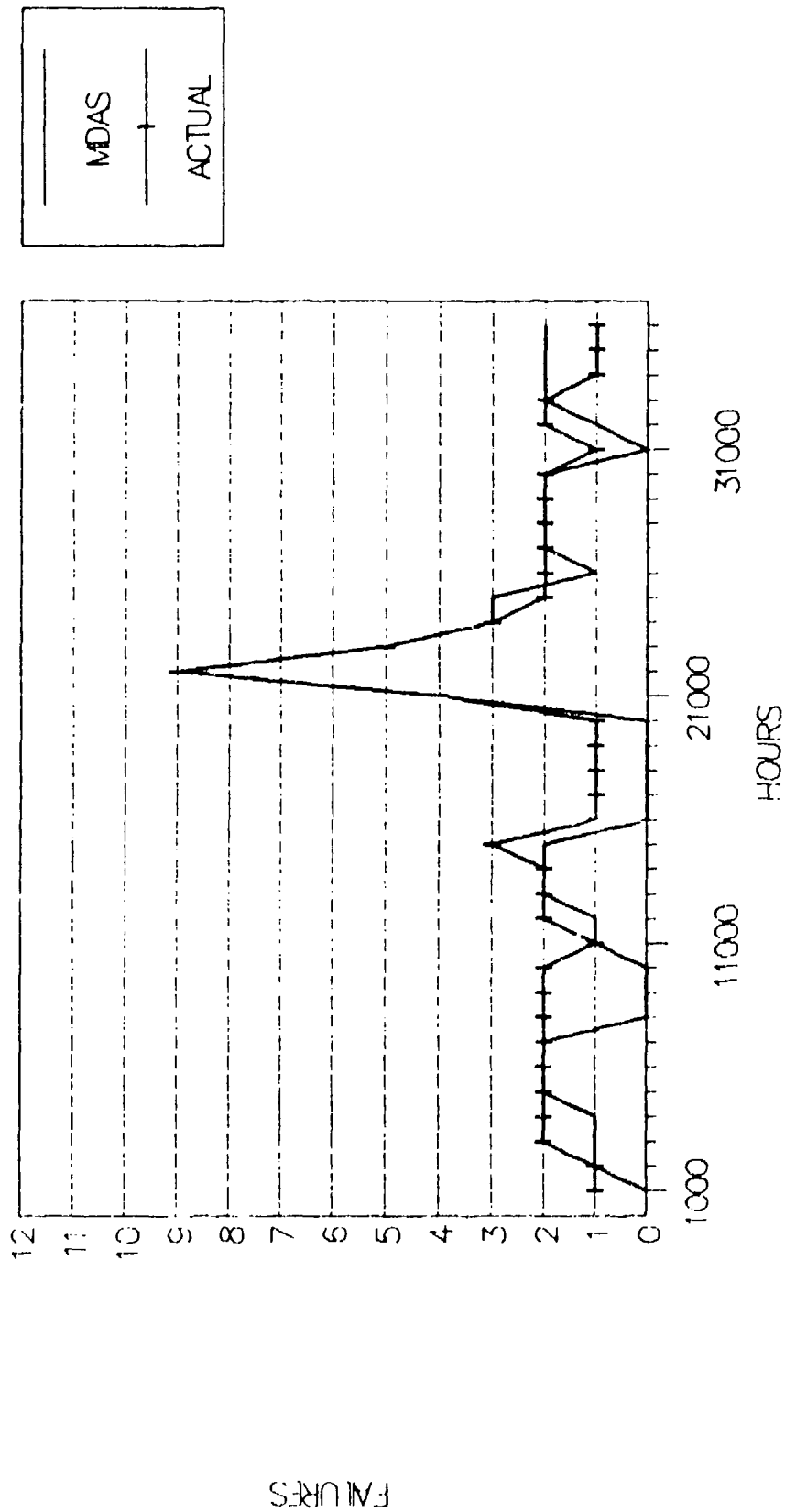
# MDAS VS ACTUAL

MTBF = 500 HOURS



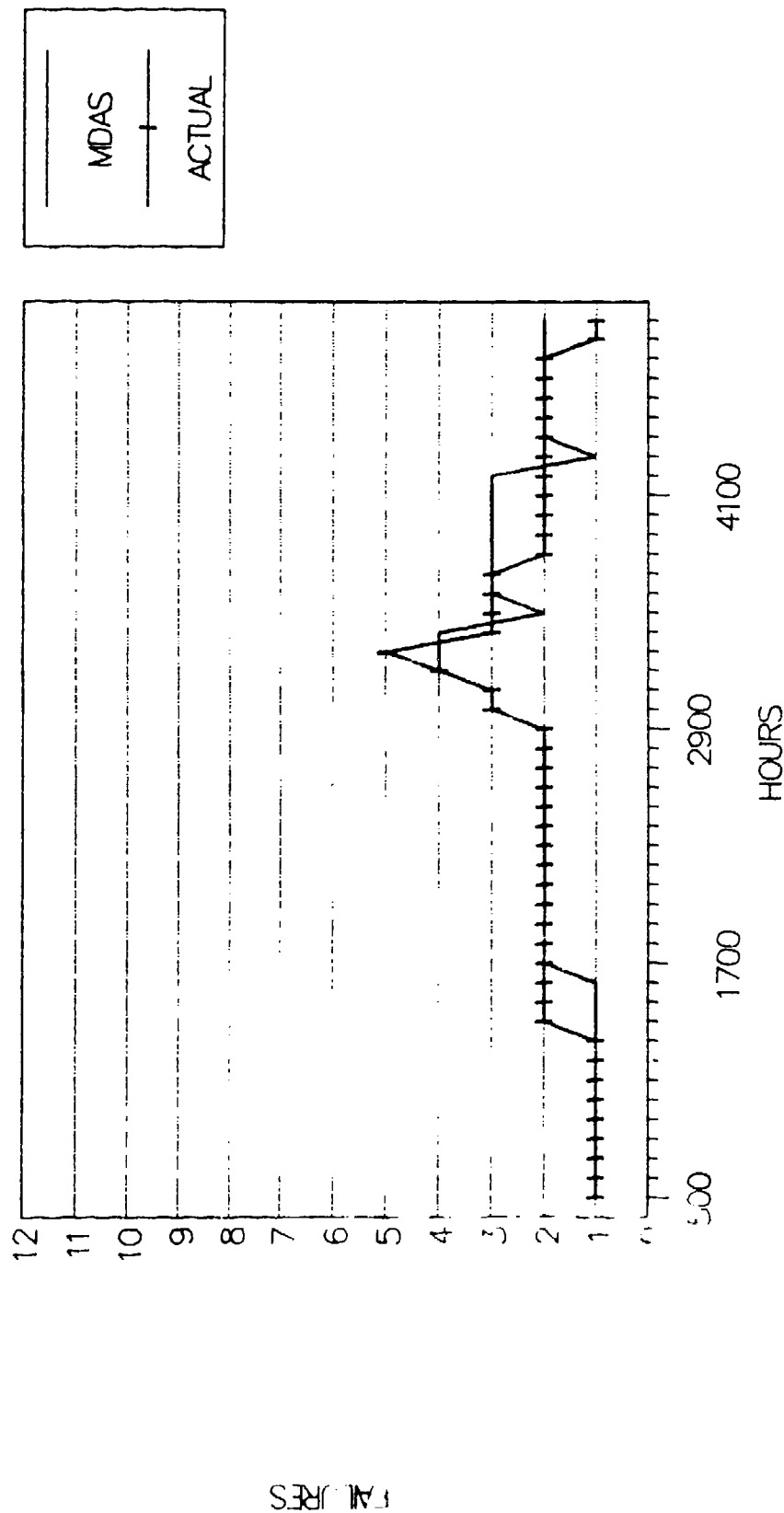
# MDAS VS ACTUAL FR

MTBF = 1000 HRS



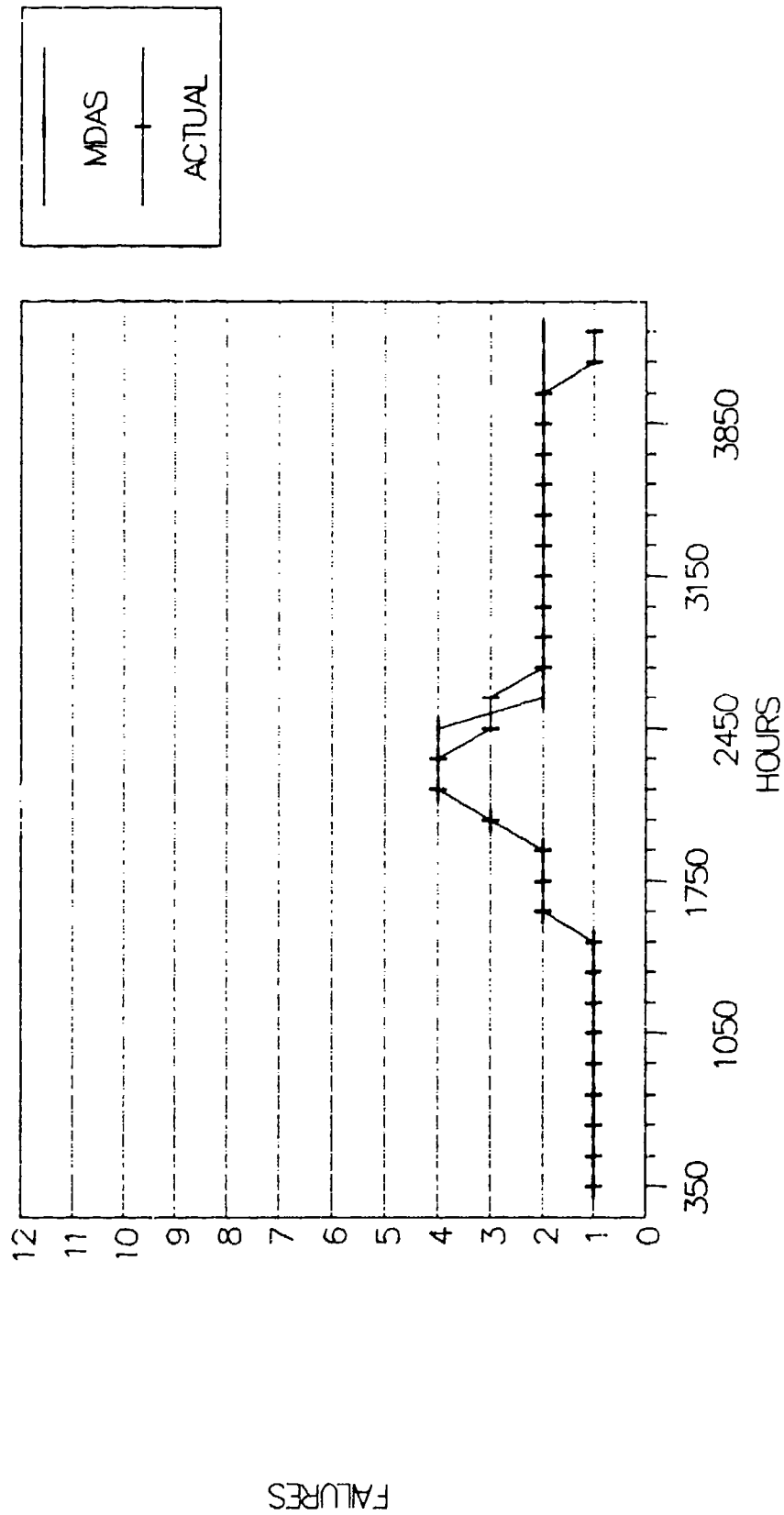
# MDAS VS ACTUAL FR

MTBF = 500



# MDAS VS ACTUAL FR

MTBF = 350 HRS



END

FILMED

3-90

DTIC